Ninth International Conference on Domain Decomposition Methods

Ninth International Conference on Domain Decomposition Methods

Edited by PETTER E. BJØRSTAD, MAGNE S. ESPEDAL AND DAVID E. KEYES

DDM.org

Dedicated to the memory of Jon Brækhus 1947-1997.
Substructures and Schur complements filled his professional life.

Contents

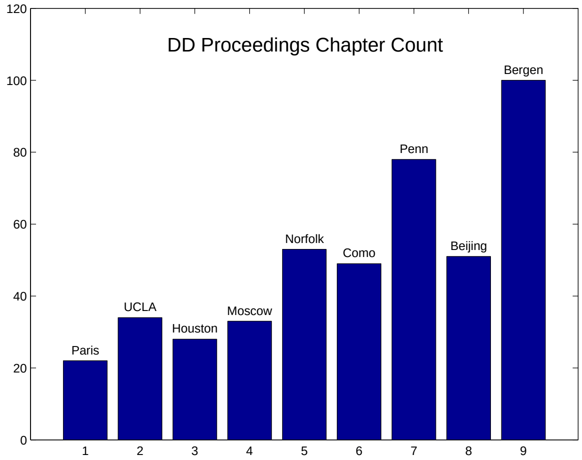
III Implementation tools 627

IV Applications 671

Preface

This volume captures about three-fourths of the proceedings of the Ninth International Conference on Domain Decomposition Methods, which was hosted by the University of Bergen in the resort village of Ullensvang, Norway, June 3–8, 1996. Approximately 180 mathematicians, engineers, physical scientists, and computer scientists from 21 countries came to this annual gathering.

Since three parallel sessions were employed at the conference in order to accommodate as many presenters as possible, attendees and non-attendees alike may turn to this volume to keep up with the diversity of subject matter that the umbrella "domain decomposition" inspires throughout the community. Its contributors are to be commended for their efforts to write for a diverse audience while staying within eight pages. Page quotas are essential to accommodate by far the largest title count in the nine-volume history of the conference.



The interest of so many authors in meeting the editorial demands and page limitations of this proceedings volume resoundingly resolves the annual and proper question of whether the common thread of domain decomposition is sufficient to justify an annual conference. It may be observed that the percentage of contributions advancing new theorems has gradually fallen from the earliest volumes, suggesting that available algebraic and function-theoretic foundations have largely been uncovered. (Perhaps there will be new graph-theoretic contributions, or infusions from other areas of mathematics in the future. In addition, we can expect relaxation of hypotheses to continue extending the theory to less ideal problems.) Meanwhile, the variety of algorithms and the variety of problems to which they are applied continue to grow, and the total number of contributions has been increasing dramatically. "Divide and conquer" may be the most basic of algorithmic paradigms, but theoreticians and practitioners alike are still seeking — and finding — incrementally more effective forms, and value the interdisciplinary forum provided by this proceedings series.

Besides inspiring elegant theory, domain decomposition methodology satisfies the architectural imperatives of high-performance computers better than methods operating only on the finest scale of the discretization (with no hierarchy) *and*, seemingly, better than methods operating simultaneously on all scales (with many levels of hierarchy). These imperatives include: spatial data locality, temporal data locality, reasonably small communication-to-computation ratios, and reasonably infrequent process synchronization (measured by the number of useful floating-point operations performed between synchronizations). Spatial data locality refers to the proximity of the addresses of successively used elements, and temporal data locality refers to the proximity in time of successive references to a given element. Spatial and temporal locality are both enhanced when a large computation based on nearest-neighbor updates is processed in contiguous blocks. On cache-based computers, subdomain blocks may be tuned for workingset sizes that reside in cache. On message-passing or cache-coherent nonuniform memory access (cc-NUMA) parallel computers, the concentration of gridpoint-oriented computations — proportional to subdomain volume — between external stencil edge-oriented communications — proportional to subdomain surface area, combined with a synchronization frequency of at most once per volume computation, gives domain decomposition excellent parallel scalability on a per iteration basis, provided only that the number of points per subdomain is not allowed to go below some problem-dependent and machine-dependent minimum in the scaling. In view of these important architectural advantages for domain decomposition methods, it is fortunate, indeed, that mathematicians studied the convergence behavior aspects of the subject in advance of the commercial arrival of these architectures, and showed how to endow domain decomposition iterative methods with some measure of algorithmic scalability, as well.

Domain decomposition has proved to be an ideal paradigm not only for execution on advanced architecture computers, but also for the development of reusable, portable software. Since the most complex operation in a Schwarz-type domain decomposition iterative method — the application of the preconditioner — is logically equivalent in each subdomain to a conventional preconditioner applied to the global domain, software developed for the global problem can readily be adapted to the local problem, instantly presenting lots of "legacy" scientific code for to be harvested for parallel implementations. Furthermore, since the only sharing of data between subdomains in domain decomposition codes occurs in two archetypal communication operations — ghost point updates in overlapping zones between neighboring subdomains, and global reduction operations, as in forming an inner product — domain decomposition methods map readily onto optimized, standardized message-passing environments, such as MPI.

Finally, it should be noted that domain decomposition is often a natural paradigm for the modeling community. Physical systems are often decomposed into two or more contiguous subdomains based on phenomenological considerations, such as the importance or neglibility of viscosity or reactivity, or any other feature, and the subdomains are discretized accordingly, as independent tasks. This physically-based domain decomposition may be mirrored in the software engineering of the corresponding code, and leads to threads of execution that operate on contiguous subdomain blocks, which can either be further subdivided or aggregated to fit the granularity of an available parallel computer, and have the correct topological and mathematical characteristics for scalability.

Organizing the contents of an interdisciplinary proceedings is an interesting job, and our decisions will inevitably surprise a few authors, though we hope without causing offense. It is increasingly artificial to assign papers to one of the four categories of theoretical foundations, algorithmic development, parallel implementation, and applications, that are traditional for this proceedings series. Readers are encouraged not to take the primary divisions very seriously, but to trace all the connections.

These proceedings will be of interest to mathematicians, computer scientists, and applications modelers, so we project its contents onto relevant classification schemes below.

American Mathematical Society (AMS) 1991 subject classifications include:

05C85 Graph algorihms

49J20 Optimal control

65C20 Numerical simulation, modeling

65D07 Spline approximation

65F10 Iterative methods for linear systems

65F15 Eigenproblems

65M55 Multigrid methods, domain decomposition for IVPs

65N30 Finite elements, Rayleigh-Ritz and Galerkin methods, finite methods

65N35 Spectral, collocation and related methods

65N55 Multigrid methods, domain decomposition for BVPs

65R20 Integral equations

65Y05 Parallel computation

68N99 Mathematical software

Association for Computing Machinery (ACM) 1998 subject classifications include:

D2 Programming environments, reusable libraries

E1 Distributed data structures

F2 Analysis and complexity of numerical algorithms

G1 Numerical linear algebra, optimization, differential equations

G4 Mathematical software, parallel implemenations, portability

J2 Applications in physical sciences and engineering

Applications for which domain decomposition methods have been specialized in this proceedings include:

fluids Stokes, Euler, Navier-Stokes, two-phase flow, reacting flow

geophysics porous media, atmospheric transport

manufacturing processes extrusion, free surface phenomena

physics neutron diffusion, semiconductor device physics

structures thermoelasticity, nonlinear elasticity, modal analysis

wave propagation acoustics, electromagnetics

For the convenience of readers coming recently into the subject of domain decomposition methods, a bibliography of previous proceedings is provided below, along with some major recent review articles and related special interest volumes. This list will inevitably be found embarrassingly incomplete. (No attempt has been made to supplement this list with the larger and closely related literature of multigrid and general iterative methods, except for the books by Hackbusch and Saad, which have significant domain decomposition components.)

1. T. F. Chan and T. P. Mathew, *Domain Decomposition Algorithms*, Acta Numerica, 1994, pp. 61-143.
2. T. F. Chan, R. Glowinski, J. Périaux and O. B. Widlund, eds., *Proc. Second Int. Symp. on Domain Decomposition Methods for Partial Differential Equations* (Los Angeles, 1988), SIAM, Philadelphia, 1989.
3. T. F. Chan, R. Glowinski, J. Périaux, O. B. Widlund, eds., *Proc. Third Int. Symp. on Domain Decomposition Methods for Partial Differential Equations* (Houston, 1989), SIAM, Philadelphia, 1990.
4. C. Farhat and F.-X. Roux, *Implicit Parallel Processing in Structural Mechanics*, Computational Mechanics Advances **2**, 1994, pp. 1–124.
5. R. Glowinski, G. H. Golub, G. A. Meurant and J. Périaux, eds., *Proc. First Int. Symp. on Domain Decomposition Methods for Partial Differential Equations* (Paris, 1987), SIAM, Philadelphia, 1988.
6. R. Glowinski, Yu. A. Kuznetsov, G. A. Meurant, J. Périaux and O. B. Widlund, eds., *Proc. Fourth Int. Symp. on Domain Decomposition Methods for Partial Differential Equations* (Moscow, 1990), SIAM, Philadelphia, 1991.

7. R. Glowinski, J. Périaux, Z.-C. Shi and O. B. Widlund, eds., *Eighth International Conference of Domain Decomposition Methods* (Beijing, 1995), Wiley, Strasbourg, 1997.
8. W. Hackbusch, *Iterative Methods for Large Sparse Linear Systems*, Springer, Heidelberg, 1993.
9. D. E. Keyes, T. F. Chan, G. A. Meurant, J. S. Scroggs and R. G. Voigt, *Proc. Fifth Int. Conf. on Domain Decomposition Methods for Partial Differential Equations* (Norfolk, 1991), SIAM, Philadelphia, 1992.
10. D. E. Keyes, Y. Saad and D. G. Truhlar, eds. *Domain-based Parallelism and Problem Decomposition Methods in Science and Engineering*, SIAM, Philadelphia, 1995.
11. D. E. Keyes and J. Xu, eds. *Proc. Seventh Int. Conf. on Domain Decomposition Methods for Partial Differential Equations* (PennState, 1993), MS, Providence, 1995.
12. P. Le Tallec, *Domain Decomposition Methods in Computational Mechanics*, Computational Mechanics Advances **2**, 1994, pp. 121–220.
13. A. Quarteroni, J. Périaux, Yu. A. Kuznetsov and O. B. Widlund, eds., *Proc. Sixth Int. Conf. on Domain Decomposition Methods in Science and Engineering* (Como, 1992), AMS, Providence, 1994.
14. Y. Saad, *Iterative Methods for Sparse Linear Systems* PWS, Boston, 1996.
15. B. F. Smith, P. E. Bjørstad and W. D. Gropp, *Domain Decomposition: Parallel Multilevel Algorithms for Elliptic Partial Differential Equations*, Cambridge Univ. Press, Cambridge, 1996.
16. J. Xu, *Iterative Methods by Space Decomposition and Subspace Correction*, SIAM Review **34**, 1991, pp. 581-613.

We also mention the homepage for domain decomposition on the World Wide Web, `www.ddm.org`, voluntarily maintained with professional skill by Tor Erling Bjørstad. This site features links to conference, bibliographic, and personal information pertaining to domain decomposition, internationally. In particular, there the reader will find a list with contact information to the authors of all 100 chapters of this book.

The technical direction of the Ninth International Conference on Domain Decomposition Methods in Scientific and Engineering Computing was provided by a scientific committee consisting of: Petter E. Bjørstad, James H. Bramble, Tony F. Chan, Peter J. Deuflhard, Roland Glowinski, David E. Keyes, Yuri A. Kuznetsov, Jacques Périaux, Alfio Quarteroni, Zhong-Ci Shi, Olof B. Widlund, and Jinchao Xu.

Local organization was undertaken by the following members of the faculty and staff at the University of Bergen: Petter E. Bjørstad, Merete Sofie Eikemo, Magne Espedal, Randi Moe, and Synnøve Palmstrøm.

The scientific and organizing committees, together with all attendees, are grateful to the following agencies, organizations, corporations, and departments for their financial and logistical support of the conference: The Norwegian Research Council, Statoil, Norsk Hydro, Sun Microsystems and Silicon Graphics.

It has turned out that the goals of traditional publishers (of proceedings) and the key objectives of the DDM proceedings as seen by the International Scientific Committee have become more and more orthogonal. We encourage broad participation and a complete proceeding showing the breath of contributions to the conference. The rapid growth of the Internet for dissemination of papers and the need to publish the proceedings in a more timely manner have led to the conclusion that the DDM proceedings shall be published directly by DDM.org starting with DD9 and DD11. (DD10 was published by AMS.) This is the first proceedings from the International Conference on Domain Decomposition Methods that is published in this way, by DDM.org, the established non-profit entity governed by the International Scientific Committee. The proceedings are freely available on the WEB page www.ddm.org as well as in book format. The editors are very grateful to Ole Arntzen and Jeremy Cook at the University of Bergen for their assistance with adapting the Latex macros to use in source-to-camera-ready preparation of the manuscript. Two distributed rounds of editing, with thanks to dozens of anonymous referees, and unforeseen technical difficulties, have delayed the release of these proceedings, but made them more worth the wait.

Our families graciously forsook much time together for this collection and are trusting, as are we, in a useful shelf life.

Petter E. Bjørstad
Bergen, Norway

Magne S. Espedal
Bergen, Norway

David E. Keyes
Hampton, Virginia, USA

January 1998

Part I

Theoretical foundations

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

1

Stabilization Techniques for Domain Decomposition Methods with Non-Matching Grids

F. Brezzi, L. P. Franca, D. Marini and A. Russo

1 Introduction

The use of domain decomposition methods with non-matching grids is becoming increasingly popular. In particular, its use is recommended when the splitting into subdomains is dictated by physical and/or geometrical reasons rather than merely by computational ones. Without underestimating the relevance of this latter group of applications (which can be extremely important and even crucial in a number of practical cases), we shall concentrate on the former one. To fix ideas, let us consider a "toy-problem" which will show well enough what we have in mind without using too heavy notation. Suppose therefore that we have a domain $\Omega =]-1,1[\times]0,1[$ split into $\Omega_1 =]-1,0[\times]0,1[$ and $\Omega_2 =]0,1[\times]0,1[$. In order to solve the problem, say,

$$-\Delta u \;=\; f \qquad \text{in } \Omega, \qquad u \;=\; 0 \qquad \text{on } \partial\Omega, \tag{1.1}$$

we decompose separately Ω_1 and Ω_2 by means of two finite element grids $\mathcal{T}_h^1$ and $\mathcal{T}_h^2$ respectively, and we want to approximate (for $i = 1,2$) u^i (restriction of u to Ω_i) by u_h^i, continuous and piecewise linear on the grid $\mathcal{T}_h^i$. Clearly, on the interface $\Gamma = \{0\}\times]0,1[$ we have two 1-d decompositions, induced by $\mathcal{T}_h^1$ and $\mathcal{T}_h^2$, which, in general, do not match. A typical solution to this (as in the mortar method [Mar90]) is to choose one of the two, say $\mathcal{T}_h^2$, and require that $u^1_{h|\Gamma}$ match $u^2_{h|\Gamma}$ only in some weak sense, with the use of suitable Lagrange multipliers. (In the mortar method terminology, the nodes of $\mathcal{T}_h^2{}_{|\Gamma}$ will be "masters" and the nodes of $\mathcal{T}_h^1{}_{|\Gamma}$, "slaves".)

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

However, in certain cases, it can be useful to choose a *third* 1-d decomposition on Γ, (say $\mathcal{T}_h^3(\Gamma)$ or simply $\mathcal{T}^\Gamma$) and have both the $\mathcal{T}_h^1{}_{|\Gamma}$ and $\mathcal{T}_h^2{}_{|\Gamma}$ nodes as "slaves". An example where this approach can be convenient is when both $\mathcal{T}_h^1{}_{|\Gamma}$ and $\mathcal{T}_h^2{}_{|\Gamma}$ are non uniform (being dictated by approximation problems that might occur in Ω_1 and Ω_2, or by self-adaptive procedures that have been used in both subdomains), but a uniform grid on Γ is recommended in order to apply a better preconditioner on the final interface problem. This suggests the use of two different Lagrange multipliers, one for matching u_h^1 with u_h^Γ, and the other one for matching u_h^2 with u_h^Γ, where, obviously, we denoted by u_h^Γ the discretization of $u_{|\Gamma}$. As it is well known, this requires suitable inf-sup conditions (see e.g. [GPP96]) to be fulfilled, one on each side of Γ. Recently, an intensive study has been carried out in order to avoid this type of inf-sup conditions by adding of suitable stabilizing terms, thus allowing more freedom in the choice of grids and multipliers (see e.g. [AG93, GG95]). In turn, in different contexts, these techniques have been reinterpreted and/or improved as the addition-elimination of suitable bubble functions to the finite element spaces in use (see e.g. [Pes72, Glo84]).

In this paper, we present a new way for stabilizing Dirichlet problems with Lagrange multipliers for the particular case where u is approximated by a piecewise linear continuous function, and the Lagrange multipliers are approximated by piecewise constant functions on a nonmatching grid. Our stabilization is made by adding suitable bubble functions only on the triangles having an edge on the boundary. It is interesting to note that elimination of the bubbles by static condensation leads to a scheme very similar to that introduced a long time ago by Nitsche [DW95] and recently reproposed and analyzed in [Osw95].

For the sake of simplicity, we shall only discuss a single-domain problem. The extension to many subdomains can then be carried out by means of the usual coupling procedures (Dirichlet-Dirichlet or Neumann-Neumann or something else).

The organization of the paper is the following. In Sect. 2 we present the single-domain problem, where the Dirichlet condition is imposed via Lagrange multipliers. In Sect. 3 we discuss its discretization with nonmatching grids and the bubble stabilization. In Sect. 4 we show that it is possible to eliminate *both* bubbles and Lagrange multipliers, thus obtaining a scheme that is easy to implementation and that strongly resembles the one discussed in [DW95, Osw95]. If needed, the Lagrange multipliers can be recovered by a simple and economical post-processing. This will be useful in a true domain decomposition situation, in order to carry out the iterative procedure.

2 The Single Domain Problem

In order to introduce our stabilization technique we shall consider a problem on a single domain, thinking of it as one of the subdomains. Always referring for simplicity to the global problem (1.1), at each step of the domain decomposition procedure we have to solve, in each subdomain, a problem of the type

$$-\Delta u \;=\; f \qquad \text{in } \Omega, \qquad u \;=\; g \qquad \text{on } \partial\Omega =: \Gamma, \tag{2.2}$$

where Ω is now the subdomain under consideration (that we assume to be a polygon), and g denotes any continuous function which, eventually, should be the value of the solution of (1.1) on $\partial\Omega \equiv$ interface between subdomains. By enforcing the boundary conditions in (2.1) with Lagrange multipliers [Ben95b], the variational formulation of (2.1) reads

$$\left\{\begin{array}{lll} \text{Find } u \in V,\ \lambda \in M \text{ such that} & & \\ \quad \int_\Omega \underline{\nabla} u \cdot \underline{\nabla} v \, dx - \int_\Gamma \lambda v \, ds & = \int_\Omega f v \, dx & \forall v \in V, \\ \qquad\qquad\qquad\qquad \int_\Gamma u\mu \, ds & = \int_\Gamma g\mu \, ds & \forall \mu \in M, \end{array}\right. \tag{2.3}$$

where λ is the multiplier, and V and M are the spaces

$$V := H^1(\Omega), \qquad M := H^{-1/2}(\Gamma)$$

with their usual norms (see [Ben95a]). With this choice for V and M, the abstract theory applies (see [GPP96]) so that problem (2.2) has a unique solution (u, λ), verifying

$$\left\{\begin{array}{rcll} -\Delta u & = & f & \text{in } \Omega \\ \lambda & = & \frac{\partial u}{\partial n} & \text{on } \Gamma \\ u & = & g & \text{on } \Gamma. \end{array}\right. \tag{2.4}$$

The usual finite element approximation of (2.2) would be to choose a decomposition $\mathcal{T}^u$ of Ω for discretizing the u variable, and take as a decomposition of Γ for the λ variable the restriction of $\mathcal{T}^u$ to Γ. Next, finite element spaces verifying the Inf-Sup condition can easily be constructed in many ways. This cannot be done in our case. Actually, in order that the discretization of (2.2) mimic the situation occurring in the domain decomposition procedure, we have to assume that the decompositions for u and g are given by $\mathcal{T}^u$ and $\mathcal{T}^g$, which do not match. Consequently, we have to introduce another decomposition of Γ, say $\mathcal{T}^\lambda$, for dealing with the multipliers λ and μ. This decomposition cannot be chosen arbitrarily, since it has to guarantee some Inf-Sup condition between the λ's and the g's, and therefore either has to coincide with $\mathcal{T}^g$ or depend on it strongly. More precisely, $\mathcal{T}^\lambda$ can be chosen finer than $\mathcal{T}^g$ without violating the Inf-Sup condition between the variables μ and the interface variables g, but it can never be coarser. In the next section we shall deal with this problem.

3 Discretization and Stabilization

Let us turn to the discretization of (2.2). Let then $\mathcal{T}_H^u$ be a decomposition of Ω into triangles $\{T\}$, H being the mesh size, and let $\mathcal{T}_h^\lambda$ be a decomposition of Γ into intervals I, h being the mesh size. We define

$$V_H = \{v \in H^1(\Omega) : \ v_{|T} \in P_1(T) \ \forall T \in \mathcal{T}_H^u\}, \tag{3.5}$$

$$M_h = \{\mu \in L^2(\Gamma) : \ \mu_{|I} \in P_0(I) \ \forall I \in \mathcal{T}_h^\lambda\}. \tag{3.6}$$

We now look for an approximate solution (u_H, λ_h) of (2.2), with $u_H \in V_H$, and $\lambda_h \in M_h$. As already pointed out, the two decompositions $\mathcal{T}_H^u$ and $\mathcal{T}_h^\lambda$ are not compatible, that is, the decomposition $\mathcal{T}_H^u$ generates a decomposition of Γ which is, in general, different from the decomposition $\mathcal{T}_h^\lambda$ of Γ. Our first step will then be to relate the two decompositions of Γ, the second step will consist in the introduction of the bubble functions, and the final step will be to analyze the stabilized problem.

1^{st} *step - Generation of a new decomposition.*

We create a new decomposition of Γ, say $\widetilde{\mathcal{T}}_h^\lambda$, by merging the two decompositions $\mathcal{T}_H^u$ and $\mathcal{T}_h^\lambda$, i.e., we add to $\mathcal{T}_h^\lambda$ the nodes of $\mathcal{T}_H^u$ belonging to Γ. In doing this, it may occur that some of the nodes of $\widetilde{\mathcal{T}}_h^\lambda$ get too close to each other, thus complicating the analysis of our procedure. To avoid this we may proceed as follows: when the distance between two nodes of $\widetilde{\mathcal{T}}_h^\lambda$ is less than or equal to some tolerance, one of the two nodes is eliminated. This can be easily done by slightly changing either the $\mathcal{T}_H^u$ or the $\mathcal{T}_h^\lambda$ decomposition, so that the two nodes become coincident. In other words, we are making the following assumption: for every triangle T in $\mathcal{T}_H^u$ having an edge E on the boundary, let H_T be the diameter of T, and let h_T be the smallest length of the intervals of $\widetilde{\mathcal{T}}_h^\lambda$ belonging to E. We assume that there exists a constant γ independent of the decompositions, such that

$$h_T \geq \gamma H_T. \tag{3.7}$$

2^{nd}*step - Introduction of the bubbles.*

We add to the discretization of u as many bubble functions as the intervals of $\widetilde{\mathcal{T}}_h^\lambda$. More precisely, we proceed as follows. Let T be a triangle having an edge on Γ. Let T' be such an edge; in general, we will have a situation of the type $T' = \cup I_k,\ I_k \in \widetilde{\mathcal{T}}_h^\lambda$ and, accordingly, $T = \cup T_k$ (see Fig. 1 as an example). We call *bubble* a function $b_k \in H^1(\Omega)$ such that $supp(b_k) \subset T_k$, and $\int_{I_k} b_k\, ds \neq 0$. (See Fig. 2). In order to have uniform estimates, we need however that the bubbles have "similar" shape. For that, let $\hat{T}$ be the reference triangle: $\hat{T} = \{(\xi,\eta) :\ 0 \leq \xi \leq 1,\ 0 \leq \eta \leq 1-\xi\}$, and let $\hat{b}$ be a function in $H^1(\hat{T})$, with $\hat{b} = 0$ on the edges $\xi = 0$ and $\eta = 0$, and $\int_{\partial\hat{T}} \hat{b}\, ds \neq 0$. (As a simple example, we can take $\hat{b}(\xi,\eta) = \xi\eta$. Many other choices are possible, and the optimal shape of $\hat{b}$ is still under investigation.) Our bubble b_k will then be given by $b_k(x,y) = \hat{b}(\xi,\eta)$ under the affine mapping $(\xi,\eta) \to (x,y)$ from $\hat{T}$ to T_k which maps the edge $\eta = 1-\xi$ on the boundary edge I_k.

3^{rd}*step - The stabilized problem.*

Let B_h be the space spanned by the bubbles introduced above. We then write the new discrete problem with V_H replaced by

$$\widetilde{V}_H := V_H \oplus B_h, \tag{3.8}$$

and M_h replaced by

$$\widetilde{M}_h \ = \ \{\mu \in L^2(\Gamma) :\ \mu_{|I} \in P_0(I)\ \forall I \in \widetilde{\mathcal{T}}_h^\lambda\}. \tag{3.9}$$

Figure 1

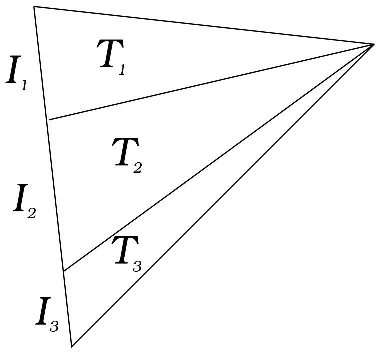
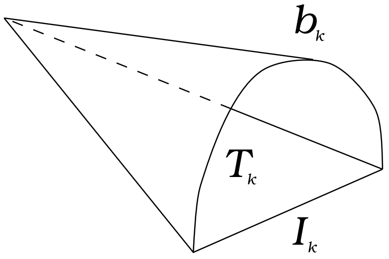


Figure 2



The approximate problem now reads

$$\begin{cases} \text{Find } u_H \in \widetilde{V}_H, \ \lambda_h \in \widetilde{M}_h \text{ such that} \\ \int_\Omega \underline{\nabla} u_H \cdot \underline{\nabla} v_H \, dx - \int_\Gamma \lambda_h v_H \, ds \quad = \int_\Omega f v_H \, dx \qquad \forall v_H \in \widetilde{V}_H, \\ \qquad\qquad\qquad\qquad \int_\Gamma \mu \, u_H \, ds \quad = \int_\Gamma g \mu \, ds \qquad \forall \mu \in \widetilde{M}_h. \end{cases} \tag{3.10}$$

Existence, uniqueness, and optimal error bounds for the solution of (3.6) will follow if we can prove the following Inf-Sup condition relating $\widetilde{V}_H$ and $\widetilde{M}_h$:

$$\begin{cases} \exists \beta > 0 \text{ independent of } h \text{ such that:} \\ \dfrac{\int_\Gamma \mu v \, ds}{||\mu||_M ||v||_V} \geq \beta \qquad \forall v \in \widetilde{V}_H, \ \forall \mu \in \widetilde{M}_h. \end{cases} \tag{3.11}$$

As the Inf-Sup condition holds for the continuous problem, (3.7) will follow from the general results of [For77], if we prove the following theorem.

Theorem 3.1 *There exists a constant C, and, for every H, a linear continuous operator $\Pi_H \ : V \longrightarrow \widetilde{V}_H$ such that*

$$\int_\Gamma (\Pi_H v - v) \mu \, ds \ = \ 0 \qquad \forall \mu \in \widetilde{M}_h, \tag{3.12}$$

and

$$||\Pi_H v||_V \leq C ||v||_V \qquad \forall v \in V. \tag{3.13}$$

Proof. We start by observing, cf. [GPP94], that it is possible to construct a linear operator $\Pi^1_H \ : \ V = H^1(\Omega) \longrightarrow V_H$ with the following properties:

$$\Pi^1_H v = v \qquad \forall v \in V_H \tag{3.14}$$

$$||\Pi^1_H v||_V \leq C ||v||_V \qquad \forall v \in V, \tag{3.15}$$

$$\forall T' \in (\mathcal{T}^u_H)_{|\Gamma} \ ||\Pi^1_H v||_{0,E} \leq C ||v||_{0,\widetilde{E}} \qquad \forall v \in V, \tag{3.16}$$

where, here and in the following, $\widetilde{E}$ is the union of the boundary edges in $\mathcal{T}_H^u$ having at least one vertex in common with E, $||v||_{0,D}$ is the norm in $L^2(D)$, $||v||_{s,D}$ the norm in $H^s(D)$, and C denotes a constant independent of the mesh size. We want to check that, for every edge E on Γ, we also have

$$||v - \Pi_H^1 v||_{0,E} \le CH_T^{1/2}||v||_{1/2,\widetilde{E}}. \tag{3.17}$$

For this, using interpolation theory (see [Ben95a, DSW96]) and (3.12), we only need to show that, for all v in $H^1(\widetilde{E})$, we have

$$||v - \Pi_H^1 v||_{0,E} \le CH_T||v||_{1,\widetilde{E}}, \tag{3.18}$$

which easily follows from (3.12) and (3.10) by the following standard argument:

$$\begin{aligned} ||v - \Pi_H^1 v||_{0,E} &\le \inf_p ||(v-p) - \Pi_H^1(v-p)||_{0,E} \\ &\le C \inf_p ||v-p||_{0,\widetilde{E}} \le CH_T||v||_{1,\widetilde{E}}, \end{aligned} \tag{3.19}$$

where the infimum is taken over the polynomials p of degree ≤ 1 in $\widetilde{E}$. Then, define another linear continuous operator $\Pi_h^2 : V \longrightarrow B_h$ as

$$\int_\Gamma (\Pi_h^2 v - v)\mu \, ds \ = \ 0 \qquad \forall \mu \in \widetilde{M_h}. \tag{3.20}$$

It can be proved that Π_h^2 is uniquely defined by (3.16), and verifies

$$||\Pi_h^2 v||_{0,T} \le CH_T^{1/2}||v||_{0,E} \qquad \forall T \in \mathcal{T}_H^u, \tag{3.21}$$

$$||\Pi_h^2 v||_{1,T} \le Ch_T^{-1}||\Pi_h^2 v||_{0,T} \qquad \forall T \in \mathcal{T}_H^u. \tag{3.22}$$

Finally, define Π_H as

$$\Pi_H v := \Pi_H^1 v + \Pi_h^2(v - \Pi_H^1 v) \qquad v \in V. \tag{3.23}$$

It is immediate to check that Π_H is linear and verifies (3.8), since, from (3.19), (3.16) we have

$$\int_\Gamma (v - \Pi_H v)\mu \, ds \ = \ \int_\Gamma \Big((v - \Pi_H^1 v) - \Pi_h^2(v - \Pi_H^1 v) \Big) \mu \, ds \ = \ 0 \qquad \forall \mu \in \widetilde{M_h}. \tag{3.24}$$

It remains to prove that Π_H verifies (3.9). We first remark that $\Pi_H v = \Pi_H^1 v$ in all triangles T that do not have edges belonging to Γ. For the remaining triangles, using (3.18)-(3.17), and (3.13) gives

$$\begin{aligned} ||\Pi_h^2(v - \Pi_H^1 v)||_{1,T} &\le Ch_T^{-1}H_T^{1/2}||v - \Pi_H^1 v||_{0,E} \\ &\le Ch_T^{-1}H_T||v||_{1/2,\widetilde{E}}, \end{aligned} \tag{3.25}$$

so that, from the definition (3.19), using (3.11) and (3.21) we have

$$\begin{aligned} ||\Pi_H v||_V &\le C \ ||\Pi_H^1 v||_V + \Big(\textstyle\sum_E ||\Pi_h^2(v - \Pi_H^1 v)||_{1,T}^2 \Big)^{1/2} \\ &\le C \ \Big(||v||_V + (\textstyle\sum_E h_T^{-2} H_T^2 ||v||_{1/2,\widetilde{E}}^2)^{1/2} \Big) \\ &\le C||v||_V, \end{aligned} \tag{3.26}$$

where, in the last inequality, we used (3.3) and the fact that

$$\sum_E ||v||^2_{1/2,\tilde{E}} \leq 3||v||^2_{1/2,\Gamma} \leq C||v||^2_V. \tag{3.27}$$

4 Interpretation of the Scheme

We will show in this section that the approximation (3.6) is directly related to Nitsche's scheme recently analyzed in Stenberg [Osw95]. For that, we rewrite (3.6) using the splitting (3.4) for trial and test functions in $\widetilde{V}_H$

$$u_H = u + \beta,\ v_H = v + b,\ u, v \in V_H,\ \beta, b \in B_h, \tag{4.28}$$

and we obtain

$$\left\{ \begin{array}{lcll} \text{Find } u \in V_H,\ \beta \in B_h,\ \lambda_h \in \widetilde{M}_h \text{ such that} & & & \\ \int_\Omega (\underline{\nabla}\, u + \underline{\nabla}\beta) \cdot \underline{\nabla}\, v\, dx - \int_\Gamma \lambda_h v\, ds & = & \int_\Omega f v\, dx & \forall v \in V_H, \\ \int_\Omega (\underline{\nabla}\, u + \underline{\nabla}\beta) \cdot \underline{\nabla} b\, dx - \int_\Gamma \lambda_h b\, ds & = & \int_\Omega f b\, dx & \forall b \in B_h, \\ \int_\Gamma \mu\, (u + \beta)\, ds & = & \int_\Gamma g \mu\, ds & \forall \mu \in \widetilde{M}_h. \end{array} \right. \tag{4.29}$$

Let us point out that, by construction, B_h and $\widetilde{M}_h$ have the same dimension, say NB. As a basis in B_h it is natural to use the functions $\{b_k\}$ defined in the previous Section (2^{nd} *step*), while a natural basis in $\widetilde{M}_h$ will be given by the functions $\mu_k =$ *the characteristic function of* I_k, for $k = 1, .., NB$. Then, we can write

$$\beta = \sum_k \beta_k b_k, \qquad \lambda_h = \sum_k \lambda_k \mu_k. \tag{4.30}$$

From the third equation of (4.2) we can derive the coefficients β_k in terms of the linear unknown u. Taking $\mu = \mu_k$ we have

$$\beta_k = \int_{I_k} (g - u)\, ds \Big/ \int_{I_k} b_k\, ds \qquad \forall k. \tag{4.31}$$

From the second equation of (4.2), taking $b = b_k$ we can express the λ_k's in terms of u and β_k

$$\begin{array}{rcll} \lambda_k & = & (\int_{T_k} \underline{\nabla}\, u \cdot \underline{\nabla} b_k\, dx + \beta_k \int_{T_k} |\underline{\nabla} b_k|^2\, dx - \int_{T_k} f b_k\, dx) / \int_{I_k} b_k\, ds & \\ & = & (\int_{I_k} b_k u_{/n}\, ds + \beta_k \int_{T_k} |\underline{\nabla} b_k|^2\, dx - \int_{T_k} f b_k\, dx) / \int_{I_k} b_k\, ds & \forall k \end{array} \tag{4.32}$$

where we have integrated the first integral by parts, and where $u_{/n}$ denotes the outward normal derivative of u. Using (4.3), the first equation of (4.2) becomes

$$\int_\Omega \underline{\nabla}\, u \cdot \underline{\nabla}\, v\, dx + \sum_k \beta_k \int_{I_k} b_k v_{/n}\, ds - \sum_k \lambda_k \int_{I_k} v\, ds \ = \ \int_\Omega f v\, dx \qquad \forall v \in V_H, \tag{4.33}$$

where again we have integrated the second integral by parts. From (4.4) and $v_{/n} =$ *constant* on I_k we have

$$\sum_k \beta_k \int_{I_k} b_k v_{/n}\, ds = \sum_k \int_{I_k} (g-u) v_{/n}\, ds = \int_\Gamma (g-u) v_{/n}\, ds. \tag{4.34}$$

Setting

$$C_k = \int_{T_k} |\underline{\nabla} b_k|^2\, dx \Big/ \left(\int_{I_k} b_k\, ds \right)^2, \tag{4.35}$$

we deduce from (4.5)

$$\sum_k \lambda_k \int_{I_k} v\, ds = \sum_k \int_{I_k} v u_{/n}\, ds + \sum_k C_k \left(\int_{I_k} (g-u)\, ds \right) \int_{I_k} v\, ds - F(v), \tag{4.36}$$

where, for the sake of simplicity, we set

$$F(v) \;=\; \sum_k \Big(\int_{T_k} f b_k\, dx \Big) \Big(\int_{I_k} v\, ds \Big) \Big/ \Big(\int_{I_k} b_k\, ds \Big). \tag{4.37}$$

The second integral in the right-hand side of (4.9) can be rewritten by using the mean value $\overline{v}$ of v on I_k, leading to

$$\sum_k C_k h_k \int_{I_k} (g-u) \overline{v}\, ds \;=\; \sum_k C_k h_k \int_{I_k} (\overline{g} - \overline{u}) \overline{v}\, ds, \tag{4.38}$$

where, obviously, h_k is the length of I_k. To simplify the notation, we can also set

$$B_\Gamma(u,v) \;=\; \sum_k C_k h_k \int_{I_k} \overline{u}\,\overline{v}\, ds. \tag{4.39}$$

Substituting (4.7) and (4.9) into (4.6), and using (4.10), (4.12) we finally obtain

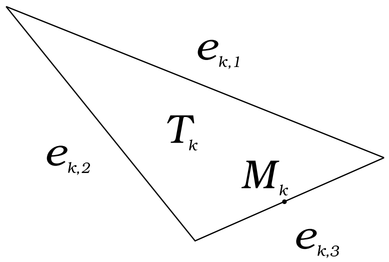
$$\left\{ \begin{array}{l} \text{Find } u \in V_H \text{ such that :} \\ \int_\Omega \underline{\nabla} u \cdot \underline{\nabla} v\, dx - \int_\Gamma v u_{/n}\, ds - \int_\Gamma u v_{/n}\, ds + B_\Gamma(u,v) = \\ \int_\Omega f v\, dx - \int_\Gamma g v_{/n}\, ds + B_\Gamma(g,v) - F(v) \qquad \forall v \in V_H. \end{array} \right. \tag{4.40}$$

It is interesting to compare (4.13) with Nitsche's method that, as studied in [Osw95], reads

$$\left\{ \begin{array}{l} \text{Find } u \in V_H \text{ such that :} \\ \int_\Omega \underline{\nabla} u \cdot \underline{\nabla} v\, dx - \int_\Gamma v u_{/n}\, ds - \int_\Gamma u v_{/n}\, ds + \alpha \int_\Gamma u v\, ds = \\ \int_\Omega f v\, dx - \int_\Gamma g v_{/n}\, ds + \alpha \int_\Gamma g v\, ds \qquad \forall v \in V_H, \end{array} \right. \tag{4.41}$$

where α is a positive parameter to be adjusted, typically, to be of the order of the inverse of the mesh size. As we can see, the only differences between (4.13) and (4.14) are: i) the use of $B_\Gamma(u,v)$ (defined in (4.12)) instead of $\alpha \int_\Gamma uv\, ds$, and ii) the addition of the term $F(v)$ to the right-hand side. In what follows, we will indicate a simple way

Figure 3



for computing $B_\Gamma(u,v)$ and $F(v)$ when using quadratic bubbles, thus producing an estimate of their order of magnitude.

Let then T be a boundary triangle, and let T_k be a subtriangle as in Fig. 1. We denote by $e_{k,i},\ i = 1,2,3$ the edges of T_k, and assume $e_{k,3}$ to be the boundary edge; M_k is the midpoint of $e_{k,3}$, and the $\lambda's$ are the usual barycentric coordinates of T_k (see Fig. 3.) With this notation, the bubble is $b_k(x,y) = \lambda_1(x,y)\lambda_2(x,y)$. With usual techniques we find

$$\int_{I_k} b_k\,ds = |e_{k,3}|/6, \qquad \int_{T_k} |\underline{\nabla} b_k|^2\,dx = \frac{(\sum_{i=1}^3 |e_{k,i}|^2)}{48|T_k|}, \tag{4.42}$$

so that (4.8) becomes

$$C_k = \frac{3(\sum_{i=1}^3 |e_{k,i}|^2)}{4|T_k||e_{k,3}|^2}. \tag{4.43}$$

Since u and v are linear on $e_{k,3}$, combining (4.12) and (4.16), and noting that in this case $h_k = |e_{k,3}|$, we obtain the following expression for $B_\Gamma(u,v)$

$$B_\Gamma(u,v) = \frac{3}{4}\sum_{k=1}^{NB} \frac{(\sum_{i=1}^3 |e_{k,i}|^2)}{|T_k|} u(M_k)v(M_k). \tag{4.44}$$

Notice that, when g is used instead of u, the value $u(M_k)$ has to be replaced by the mean value of g in I_k. We also point out that, comparing (4.17) with (4.14), we see that our method corresponds to choosing, in each I_k, a value of α of the order of H_T/h_k^2.

We now turn to the computation of the term $F(v)$, assuming that f is constant in T_k and v is a basis function in V_H. Clearly, from (4.10) we have $F(v) = 0$ if v is associated with an internal vertex of $\mathcal{T}_H^u$. Otherwise, a simple computation shows that

$$F(v) = \sum_{k=1}^{NB} f_k \frac{|T_k|}{2|e_{k,3}|} \int_{I_k} v\,ds. \tag{4.45}$$

In addition, it can easily be checked that

$$\frac{|T_k|}{2|e_{k,3}|}\int_{I_k} v\,ds = \frac{|T_k|v(M_k)}{2} = \frac{3}{4}\int_{T_k} v\,dx. \tag*{(4,46)}$$

Hence,

$$F(v) = \frac{3}{4}\sum_{k=1}^{NB}\int_{T_k} fv\,dx. \tag{4.47}$$

Finally, we point out that, in domain decomposition procedures, the explicit knowledge of the Lagrange multiplier λ_h in (3.6) is needed in order to update the interface unknown g during an iterative solution. With our approach, once u has been computed out of (4.13), the value of λ_h in each I_k can be easily recovered from (4.5), which gives

$$\lambda_k = (u_{/n})_{|I_k} + C_k \int_{I_k} (g-u)\,ds - f_k|T_k|/(2|e_{k,3}|). \tag{4.48}$$

5 Conclusions

The single-domain Dirichlet problem for a linear elliptic operator can be solved by the Lagrange multipliers technique, which is well suited when the boundary condition is given on a grid which does not match with the one used within the domain. If the problem with Lagrange multipliers is stabilized by boundary bubbles, it is possible (with "paper and pencil") to eliminate *a priori* both bubbles *and* Lagrange multipliers. The resulting scheme, which is quite simple to implement, results in a variant of the Nitsche's method [DW95]. As needed in domain decomposition procedures, the Lagrange multipliers can then be computed afterwards, in each subdomain, by an easy and economical post-processing.

REFERENCES

[Bab73] Babuška I. (1973) The finite element method with Lagrangian multipliers. *Numer. Math.* 20: 179–192.

[BBF93] Baiocchi C., Brezzi F., and Franca L. (1993) Virtual bubbles and the Galerkin-least-squares method. *Comput. Methods Appl. Mech. Engrg.* 105: 125–141.

[BBM92] Baiocchi C., Brezzi F., and Marini D. (1992) Stabilization of Galerkin methods and applications to domain decomposition. In Bensoussan A. and Verjus J.-P. (eds) *Future Tendencies in Computer Science, Control and Applied Mathematics*, volume 653 of *Lecture Notes in Computer Science*, pages 345–355. Springer-Verlag. Proceedings of the International Conference on the Occasion of the 25th Anniversary of INRIA, Paris, France, December 1992.

[BF91] Brezzi F. and Fortin M. (1991) *Mixed and Hybrid Finite Element Methods*, volume 15 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, New-York.

[BFHR96] Brezzi F., Franca L., Hughes T., and Russo A. (April 1–4 1996) Stabilization techniques and subgrid scales capturing. In *Proceedings of the SOTANA meeting.*

[BH92] Barbosa H. and Hughes T. J. R. (1992) Boundary Lagrange multipliers in finite element methods: error analysis in natural norms. *Numer. Math.* 62: 1–16.

[BL76] Bergh J. and Lofstrom J. (1976) *Interpolation spaces: an introduction.* Springer-Verlag.

[BMP89] Bernardi C., Maday Y., and Patera A. (1989) A new nonconforming approach to domain decompositions: The mortar element method. In H.Brezis and J.L.Lions (eds) *Nonlinear Partial Differential Equations and their Applications.* Pitman and Wiley.

[For77] Fortin M. (1977) An analysis of the convergence of mixed finite element methods. *RAIRO Anal. Numer.* (11): 341–354.

[LM72] Lions J. and Magenes E. (1972) *Non homogeneous boundary value problems and applications, I, II.* Grund. B. Springer-Verlag.

[Nit71] Nitsche J. (1970–71) Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen die keinen Randbedingungen unterworfen sind. *Abh. Math. Sem. Univ. Hamburg* 36: 9–15.

[Ste94] Stenberg R. (1994) On some techniques for approximating boundary conditions in the finite element method. Technical Report 22, Helsinki University of Technology, Laboratory for Strength of Materials.

[SZ90] Scott L. and Zhang S. (1990) Finite element interpolation of nonsmooth functions. *Math. Comp.* (54): 483–493.

2

Preconditioning in H(div) and Applications

Douglas N. Arnold, Richard S. Falk and Ragnar Winther

1 Introduction

This paper summarizes the work of [Pes72], in which we consider the solution of the system of linear algebraic equations which arises from the finite element discretization of boundary value problems in two space dimensions for the differential operator $\boldsymbol{I} - \mathbf{grad}\,\mathrm{div}$. The natural setting for the weak formulation of such problems is the space:

$$\boldsymbol{H}(\mathrm{div}) = \left\{ \boldsymbol{u} \in \boldsymbol{L}^2(\Omega) \mid \mathrm{div}\,\boldsymbol{u} \in L^2(\Omega) \right\}.$$

Let $(\,\cdot\,,\,\cdot\,)$ denote the $L^2(\Omega)$ inner product of both scalar and vector-valued functions and

$$J(\boldsymbol{u},\boldsymbol{v}) := (\boldsymbol{u},\boldsymbol{v}) + (\mathrm{div}\,\boldsymbol{u}, \mathrm{div}\,\boldsymbol{v})$$

denote the innerproduct on $\boldsymbol{H}(\mathrm{div})$. If $\boldsymbol{f} \in \boldsymbol{L}^2(\Omega)$, the weak formulation is to find $\boldsymbol{u} \in \boldsymbol{H}(\mathrm{div})$ such that for all $\boldsymbol{v} \in \boldsymbol{H}(\mathrm{div})$,

$$J(\boldsymbol{u},\boldsymbol{v}) = (\boldsymbol{f},\boldsymbol{v}).$$

This corresponds to the boundary value problem

$$(\boldsymbol{I} - \mathbf{grad}\,\mathrm{div})\boldsymbol{u} = \boldsymbol{f} \text{ in } \Omega, \qquad \mathrm{div}\,\boldsymbol{u} = 0 \text{ on } \partial\Omega.$$

Note that if $\boldsymbol{u}$ is a gradient, then $(\boldsymbol{I} - \mathbf{grad}\,\mathrm{div})\boldsymbol{u} = -\boldsymbol{\Delta}\,\boldsymbol{u} + \boldsymbol{u}$, while if $\boldsymbol{u}$ is a curl, then $(\boldsymbol{I} - \mathbf{grad}\,\mathrm{div})\boldsymbol{u} = \boldsymbol{u}$. A simple situation in which the operator $\boldsymbol{I} - \mathbf{grad}\,\mathrm{div}$ arises occurs in the computation of $\boldsymbol{u} = \mathbf{grad}\,p$, where p is the solution of the Dirichlet problem

$$-\Delta\,p + p = g \quad \text{in } \Omega, \qquad p = 0 \quad \text{on } \partial\Omega.$$

Then $\boldsymbol{u} \in \boldsymbol{H}(\mathrm{div})$ satisfies

$$J(\boldsymbol{u},\boldsymbol{v}) = -(g, \mathrm{div}\,\boldsymbol{v}) \quad \text{for all } \boldsymbol{v} \in \boldsymbol{H}(\mathrm{div}).$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Given a finite element subspace $\boldsymbol{V}_h$ of $\boldsymbol{H}(\mathrm{div})$, the natural finite element approximation scheme is: Find $\boldsymbol{u}_h \in \boldsymbol{V}_h$ such that

$$J(\boldsymbol{u}_h, \boldsymbol{v}_h) = (\boldsymbol{f}, \boldsymbol{v}_h) \qquad \text{for all } \boldsymbol{v}_h \in \boldsymbol{V}_h.$$

We shall consider the case when $\boldsymbol{V}_h$ consists of the Raviart–Thomas space of index $k \geq 0$, i.e., functions which on each triangle are of the form

$$\boldsymbol{v}(x,y) = \boldsymbol{p}(x,y) + (x,y)q(x,y), \qquad \boldsymbol{p} \in \mathcal{P}_k \times \mathcal{P}_k, \quad q \in \mathcal{P}_k,$$

(where $\mathcal{P}_k$ denotes the polynomials of degree $\leq k$) and for which $\boldsymbol{v}\cdot\boldsymbol{n}$ is continuous from triangle to triangle. The goal is to find an efficient procedure for solving the discrete linear system corresponding to this discretization, which we write as $\boldsymbol{J}_h\boldsymbol{u}_h = \boldsymbol{f}_h$. Denoting the eigenvalues of $\boldsymbol{J}_h$ by $\sigma(\boldsymbol{J}_h)$, since the spectral condition number

$$\kappa(\boldsymbol{J}_h) := \frac{\max|\sigma(\boldsymbol{J}_h)|}{\min|\sigma(\boldsymbol{J}_h)|}$$

of the operator $\boldsymbol{J}_h$ is $O(h^{-2})$, we will clearly need to precondition any standard iterative scheme if we want the number of iterations needed to achieve a given accuracy to be independent of h.

2 Preconditioning in the Abstract

Let $X_h \subset L^2$ be a finite dimensional normed vectorspace. We identify X_h and X_h^* as sets, but put the dual norm on the latter (dual with respect to the L^2 inner product). Let $\mathcal{A}_h : X_h \to X_h$ be an L^2-symmetric linear isomorphism. We suppose that X_h is endowed with an appropriate (energy) norm, i.e., we suppose that

$$\|\mathcal{A}_h\|_{\mathcal{L}(X_h,X_h^*)},\ \|\mathcal{A}_h^{-1}\|_{\mathcal{L}(X_h^*,X_h)} = O(1).$$

Given $f_h \in X_h$, we wish to solve $\mathcal{A}_h x_h = f_h$ by applying a standard iterative method such as CG or MINRES to the equation $\mathcal{B}_h\mathcal{A}_h x_h = \mathcal{B}_h f_h$, where $\mathcal{B}_h : X_h \to X_h$ is an L^2-symmetric, positive definite preconditioner. Our goal is to define $\mathcal{B}_h$ so that the action of $\mathcal{B}_h$ is easily computable and $\kappa(\mathcal{B}_h\mathcal{A}_h)$ is bounded uniformly with respect to h. Since

$$\max|\sigma(\mathcal{B}_h\mathcal{A}_h)| \leq \|\mathcal{B}_h\mathcal{A}_h\|_{\mathcal{L}(X_h,X_h)} \leq \|\mathcal{A}_h\|_{\mathcal{L}(X_h,X_h^*)}\|\mathcal{B}_h\|_{\mathcal{L}(X_h^*,X_h)}$$

and

$$\frac{1}{\min|\sigma(\mathcal{B}_h\mathcal{A}_h)|} \leq \|(\mathcal{B}_h\mathcal{A}_h)^{-1}\|_{\mathcal{L}(X_h,X_h)} \leq \|\mathcal{A}_h^{-1}\|_{\mathcal{L}(X_h^*,X_h)}\|\mathcal{B}_h^{-1}\|_{\mathcal{L}(X_h,X_h^*)}$$

$\mathcal{B}_h$ is an effective preconditioner if

$$\|\mathcal{B}_h\|_{\mathcal{L}(X_h^*,X_h)},\ \|\mathcal{B}_h^{-1}\|_{\mathcal{L}(X_h,X_h^*)} = O(1).$$

In other words, $\mathcal{B}_h$ is an effective preconditioner if it has the same mapping properties as $\mathcal{A}_h^{-1}$. Note that the energy norm, and not the detailed structure of $\mathcal{A}_h$, determine these properties. Thus to solve the problem $\boldsymbol{J}_h \boldsymbol{u}_h = \boldsymbol{f}_h$, we need to construct an efficiently computable operator $\boldsymbol{K}_h : \boldsymbol{V}_h \to \boldsymbol{V}_h$ for which

$$\|\boldsymbol{K}_h\|_{\mathcal{L}(V_h^*, V_h)}, \; \|\boldsymbol{K}_h^{-1}\|_{\mathcal{L}(V_h, V_h^*)} = O(1).$$

We will show how this can be done using domain decomposition and multigrid techniques.

3 Applications

We are interested in the operator $\boldsymbol{I} - \mathbf{grad}\,\mathrm{div}$ not for its own sake, but for its appearance in several important problems. Besides the example mentioned in the introduction, we will restrict our attention to two problems: the least squares formulation and the mixed formulation of second order scalar elliptic problems. Other examples are discussed in [Pes72]. We first discuss the least squares variational principle.

Consider the elliptic boundary value problem

$$\mathrm{div}(A\ \mathbf{grad}\, p) = g \text{ in } \Omega, \qquad p = 0 \text{ on } \partial\Omega,$$

where the coefficient matrix A is assumed measurable, bounded, symmetric, and uniformly positive definite on Ω. Introducing $\boldsymbol{u} = A\ \mathbf{grad}\, p$ leads to the first order system

$$\boldsymbol{u} - A\ \mathbf{grad}\, p = 0 \text{ in } \Omega, \quad \mathrm{div}\, \boldsymbol{u} = g \text{ in } \Omega, \quad p = 0 \text{ on } \partial\Omega.$$

The least squares variational principle characterizes the solution $(\boldsymbol{u}, p)$ as the minimizer of the functional

$$\|\boldsymbol{v} - A\ \mathbf{grad}\, q\|^2 + \|\,\mathrm{div}\, \boldsymbol{v} - g\|^2$$

over the space $\boldsymbol{H}(\mathrm{div}) \times \mathring{H}^1$, where $\|\cdot\|$ denotes the $L^2(\Omega)$ norm and $\mathring{H}^1$ denotes the subspace of functions in $H^1(\Omega)$ which vanish on the boundary of Ω. Equivalently, we have the weak formulation

$$B(\boldsymbol{u}, p; \boldsymbol{v}, q) = (g, \mathrm{div}\, \boldsymbol{v}) \quad \text{for all } (\boldsymbol{v}, q) \in \boldsymbol{H}(\mathrm{div}) \times \mathring{H}^1,$$

where

$$B(\boldsymbol{u}, p; \boldsymbol{v}, q) = (\boldsymbol{u} - A\ \mathbf{grad}\, p, \boldsymbol{v} - A\ \mathbf{grad}\, q) + (\mathrm{div}\, \boldsymbol{u}, \mathrm{div}\, \boldsymbol{v}).$$

To discretize the least squares formulation, we let $X_h = \boldsymbol{V}_h \times W_h$ be a finite-dimensional subspace of $\boldsymbol{H}(\mathrm{div}) \times \mathring{H}^1$. Then $x_h := (\boldsymbol{u}_h, p_h)$ is the minimizer over X_h of

$$\|\boldsymbol{v} - A\ \mathbf{grad}\, q\|^2 + \|\,\mathrm{div}\, \boldsymbol{v} - g\|^2,$$

or in weak form,

$$B(\boldsymbol{u}_h, p_h; \boldsymbol{v}, q) = (g, \mathrm{div}\, \boldsymbol{v}) \quad \text{for all } (\boldsymbol{v}, q) \in X_h.$$

Defining $\mathcal{A}_h : X_h \to X_h$ by $(\mathcal{A}_h x, y) = B(x, y)$ and $f_h \in X_h$ by $(f_h, (\boldsymbol{v}, q)) = (g, \mathrm{div}\, \boldsymbol{v})$, we may rewrite our problem as $\mathcal{A}_h x_h = f_h$.

The key to the convergence theory for the least squares method is the following theorem (cf. Pehlivanov, Carey, Lazarov [Mar90] and Cai, Lazarov, Manteuffel, and McCormick [AG93]).

Theorem 2.1 *The bilinear form B is an inner product on $\boldsymbol{H}(\mathrm{div}) \times \mathring{H}^1$ equivalent to the usual one.*

A direct consequence of the theorem is that $\mathcal{A}_h : X_h \to X_h$ is symmetric, positive definite and satisfies

$$\|\mathcal{A}_h\|_{\mathcal{L}(X_h, X_h^*)},\ \|\mathcal{A}_h^{-1}\|_{\mathcal{L}(X_h^*, X_h)} = O(1).$$

Thus we need a preconditioner with the opposite mapping properties. Since $X_h = \boldsymbol{V}_h \times W_h$, we can choose a block diagonal preconditioner

$$\mathcal{B}_h = \begin{pmatrix} \boldsymbol{K}_h & 0 \\ 0 & M_h \end{pmatrix},$$

where $\boldsymbol{K}_h$ is a good preconditioner in $\boldsymbol{H}(\mathrm{div})$, i.e., it maps like $\boldsymbol{J}_h^{-1} : \boldsymbol{V}_h \to \boldsymbol{V}_h$, and M_h is a good preconditioner in $\mathring{H}^1$, i.e., it maps like $\Delta_h^{-1} : W_h \to W_h$. Hence we conclude that a good preconditioner for the discrete least squares system is obtained using an $\boldsymbol{H}(\mathrm{div})$ preconditioner for the vector variable and a standard $\mathring{H}^1$ preconditioner for the scalar variable.

We next consider a mixed variational formulation of this boundary value problem. The mixed variational principle characterizes $(\boldsymbol{u}, p)$ as a saddle point of

$$\frac{1}{2}(A^{-1}\boldsymbol{v}, \boldsymbol{v}) + (q, \operatorname{div} \boldsymbol{v}) - (g, q),$$

over $\boldsymbol{H}(\mathrm{div}) \times L^2$, or, in weak form,

$$(A^{-1}\boldsymbol{u}, \boldsymbol{v}) + (p, \operatorname{div} \boldsymbol{v}) = 0 \qquad \text{for all } \boldsymbol{v} \in \boldsymbol{H}(\mathrm{div}),$$

$$(\operatorname{div} \boldsymbol{u}, q) = (g, q) \qquad \text{for all } q \in L^2.$$

Choosing $X_h = \boldsymbol{V}_h \times S_h \subset \boldsymbol{H}(\mathrm{div}) \times L^2$, we can define a discrete solution $x_h = (\boldsymbol{u}_h, p_h) \in \boldsymbol{V}_h \times S_h$ by restricting either the variational or weak formulation. This may be written $\mathcal{A}_h x_h = f_h$, with $\mathcal{A}_h : X_h \to X_h$ L^2-symmetric but indefinite, since $\mathcal{A}_h$ has the form

$$\mathcal{A}_h = \begin{pmatrix} a & b \\ b^t & 0 \end{pmatrix}.$$

The convergence of this method depends on the choice of $\boldsymbol{V}_h$ and S_h. The key hypotheses for the convergence analysis are the Brezzi conditions:

$$(A^{-1}\boldsymbol{v}, \boldsymbol{v}) \geq \gamma_1 \|\boldsymbol{v}\|_{H(\mathrm{div})} \qquad \text{for all } \boldsymbol{v} \in \boldsymbol{V}_h \text{ with } \operatorname{div} \boldsymbol{v} \perp S_h,$$

$$\inf_{q \in S_h} \sup_{\boldsymbol{v} \in \boldsymbol{V}_h} \frac{(q, \operatorname{div} \boldsymbol{v})}{\|q\| \, \|\boldsymbol{v}\|_{H(\mathrm{div})}} \geq \gamma_2.$$

These conditions are satisfied if $\boldsymbol{V}_h$ is the Raviart–Thomas space of index k and S_h the space of (discontinuous) piecewise polynomials of degree k. Brezzi's theorem states that if both hypotheses are satisfied, then $\mathcal{A}_h$ is an isomorphism and $\|\mathcal{A}_h^{-1}\|_{\mathcal{L}(X_h^*, X_h)}$ may be bounded in terms of the γ_i.

We thus base our choice of $\mathcal{B}_h$ on the discrete version of the isomorphism

$$\begin{pmatrix} A & -\mathbf{grad} \\ \operatorname{div} & 0 \end{pmatrix} : \boldsymbol{H}(\operatorname{div}) \times L^2 \to \boldsymbol{H}(\operatorname{div})^* \times L^2.$$

We again use a simple block-diagonal preconditioner, which this time takes the form

$$\mathcal{B}_h = \begin{pmatrix} \boldsymbol{K}_h & 0 \\ 0 & I \end{pmatrix},$$

where I is the identity on S_h and again $\boldsymbol{K}_h$ is a good preconditioner in $\boldsymbol{H}(\operatorname{div})$, i.e., it maps like $\boldsymbol{J}_h^{-1} : \boldsymbol{V}_h \to \boldsymbol{V}_h$.

We remark that most other work on preconditioning such mixed methods uses the alternate isomorphism

$$\begin{pmatrix} A & -\mathbf{grad} \\ \operatorname{div} & 0 \end{pmatrix} : \boldsymbol{L}^2 \times \mathring{H}^1 \to \boldsymbol{L}^2 \times H^{-1},$$

which leads to a different (and less natural) choice of preconditioner.

4 An Additive Schwarz Preconditioner for $\boldsymbol{J}_h$

We let $\mathcal{T}_H = \{\Omega_n\}_{n=0}^N$, denote the coarse mesh and $\mathcal{T}_h$ a refinement (the fine mesh). We let $\{\Omega'_n\}_{n=1}^N$ be an overlapping covering aligned with the fine mesh such that $\Omega_n \subset \Omega'_n$. We make the standard assumption of sufficient but bounded overlap. Let $\boldsymbol{V}_n$ denote the Raviart–Thomas space approximating $\boldsymbol{H}(\operatorname{div}, \Omega'_n)$ with the boundary condition $\boldsymbol{v} \cdot \boldsymbol{n} = 0$ on $\partial\Omega'_j \setminus \partial\Omega$. Let $\boldsymbol{V}_0$ denote the Raviart–Thomas approximation to $\boldsymbol{H}(\operatorname{div}, \Omega)$ using the coarse mesh.

Given $\boldsymbol{f} \in \boldsymbol{V}_h$, define $\boldsymbol{u}_n \in \boldsymbol{V}_n$ by $J(\boldsymbol{u}_n, \boldsymbol{v}) = (\boldsymbol{f}, \boldsymbol{v})$ for all $\boldsymbol{v} \in \boldsymbol{V}_n$. The additive Schwarz preconditioner is then defined by $\boldsymbol{K}_h \boldsymbol{f} := \sum_{n=0}^N \boldsymbol{u}_n$. Our main result for this domain decomposition preconditioner is the following theorem (cf. [Pes72] for the proof).

Theorem 2.2 *There exists a constant c independent of both h and H for which $\kappa(\boldsymbol{K}_h \boldsymbol{J}_h) \le c$.*

Following the theoretical framework of Dryja–Widlund [GG95] or Xu [GPP96], a critical step of the proof is the following decomposition lemma.

Lemma 2.1 *For all $\boldsymbol{v} \in \boldsymbol{V}_h$, there exist $\boldsymbol{v}_n \in \boldsymbol{V}_n$ with $\boldsymbol{v} = \sum_{n=0}^N \boldsymbol{v}_n$ and*

$$\sum_{n=0}^N \|\boldsymbol{v}_n\|^2_{H(\operatorname{div})} \le c\|\boldsymbol{v}\|_{H(\operatorname{div})}.$$

The standard proof uses a partition of unity $\{\theta_n\}_{n=1}^N$ and takes $\boldsymbol{v}_0 \in \boldsymbol{V}_0$ a suitable approximation of $\boldsymbol{v}$ and $\boldsymbol{v}_n = \boldsymbol{\Pi}_h[\theta_n(\boldsymbol{v} - \boldsymbol{v}_0)]$ with $\boldsymbol{\Pi}_h$ a suitable local projection into $\boldsymbol{V}_h$. The analysis leads to the following estimates.

$$\begin{aligned} \|\operatorname{div} \boldsymbol{v}_n\| \le\ & c\|\operatorname{div}[\theta_n(\boldsymbol{v} - \boldsymbol{v}_0)]\| \\ \le\ & c\|\mathbf{grad}\,\theta_n\|_{L^\infty}\|\boldsymbol{v} - \boldsymbol{v}_0\| + \|\theta_n\|_{L^\infty}\|\operatorname{div}(\boldsymbol{v} - \boldsymbol{v}_0)\| \\ \le\ & cH^{-1}\|\boldsymbol{v} - \boldsymbol{v}_0\| + \|\operatorname{div}(\boldsymbol{v} - \boldsymbol{v}_0)\|. \end{aligned}$$

In the standard elliptic case we bound the first term using $\|\boldsymbol{v}-\boldsymbol{v}_0\| \le CH\|\boldsymbol{v}\|_1$. However it is not true that $\|\boldsymbol{v}-\boldsymbol{v}_0\| \le CH\|\boldsymbol{v}\|_{H(\mathrm{div})}$, so this approach fails. We are able to get around this problem by using a discrete Helmholtz decomposition, which we now describe.

Let $\boldsymbol{V}_h$ denote the Raviart–Thomas space of index k, S_h the space of piecewise polynomials of degree k, and W_h the space of C^0 piecewise polynomials of degree $k+1$. Then we have the following discrete Helmholtz decomposition.

$$\boldsymbol{V}_h = \mathbf{curl}\, W_h \oplus \mathbf{grad}\, S_h,$$

where $\mathbf{grad} : S_h \to \boldsymbol{V}_h$ is defined by $(\mathbf{grad}\, s, \boldsymbol{v}) = -(s, \operatorname{div}\boldsymbol{v})$.

Returning to the decomposition lemma, we write $\boldsymbol{v} = \mathbf{curl}\, w + \mathbf{grad}\, s$ and observe that

$$\|\boldsymbol{v}\|^2_{H(\mathrm{div})} = \|\,\mathbf{curl}\, w\|^2 + \|\,\mathbf{grad}\, s\|^2_{H(\mathrm{div})}.$$

We then decompose each term separately. Since $\|\,\mathbf{curl}\, w\|_{H(\mathrm{div})} \approx \|w\|_1$, we can use the standard decomposition lemma on w to write

$$w = \sum_{j=0}^{n} w_j, \qquad \sum_{j=0}^{n} \|w_j\|_1^2 \le c\|w\|_1^2.$$

Taking curls gives us the desired result on the $\mathbf{curl}\, w$ term.

For $\boldsymbol{v} = \mathbf{grad}\, s$ and $\boldsymbol{v}_0 = \mathbf{grad}\, s_0$, where $(s_0, \boldsymbol{v}_0)$ is the mixed method approximation to $(s, \boldsymbol{v})$ in the space $S_0 \times \boldsymbol{V}_0$, we can prove using standard results from the theory of mixed finite element approximations that

$$\|\boldsymbol{v}-\boldsymbol{v}_0\| \le CH\|\boldsymbol{v}\|_{H(\mathrm{div})},$$

and conclude the proof. The key is that although the above estimate does not hold for all $\boldsymbol{v} \in \boldsymbol{V}_h$, it does hold when $\boldsymbol{v} = \mathbf{grad}\, s$.

5 V-cycle Preconditioner

We consider a nested sequence of meshes, $\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_N$, and let $\boldsymbol{V}_n$ be the Raviart-Thomas space of some fixed order subordinate to the mesh $\mathcal{T}_n$. This gives a nested sequence of spaces $\boldsymbol{V}_1 \subset \boldsymbol{V}_2 \subset \cdots \subset \boldsymbol{V}_N = \boldsymbol{V}_h$ and corresponding operators $\boldsymbol{J}_n : \boldsymbol{V}_n \to \boldsymbol{V}_n$.

We also require smoothers $\boldsymbol{R}_n : \boldsymbol{V}_n \to \boldsymbol{V}_n$ which we discuss below and the $\boldsymbol{H}(\mathrm{div})$-projection operators $\boldsymbol{P}_n : \boldsymbol{H}(\mathrm{div}) \to \boldsymbol{V}_n$. Multigrid then defines $\boldsymbol{K}_n : \boldsymbol{V}_n \to \boldsymbol{V}_n$ recursively starting with $\boldsymbol{K}_1 = \boldsymbol{J}_1^{-1}$. We shall make use of the following multigrid convergence result.

Theorem 2.3 *Suppose that for each $n = 1, 2, \dots, N$ the smoother R_n is L^2-symmetric and positive semi-definite and satisfies the conditions*

$$J([\boldsymbol{I} - \boldsymbol{R}_n\boldsymbol{J}_n]\boldsymbol{v}, \boldsymbol{v}) \ge 0$$

$$(\boldsymbol{R}_n^{-1}[\boldsymbol{I} - \boldsymbol{P}_{n-1}]\boldsymbol{v}, [\boldsymbol{I} - \boldsymbol{P}_{n-1}]\boldsymbol{v}) \le \alpha J([\boldsymbol{I} - \boldsymbol{P}_{n-1}]\boldsymbol{v}, [\boldsymbol{I} - \boldsymbol{P}_{n-1}]\boldsymbol{v}).$$

Table 1 Condition numbers for the operator $\boldsymbol{J}_h$ and for the preconditioned operator $\boldsymbol{K}_h\boldsymbol{J}_h$, and iterations counts to achieve an error reduction factor of 10^6.

level	h	elements	$\dim \boldsymbol{V}_h$	$\kappa(\boldsymbol{J}_h)$	$\kappa(\boldsymbol{K}_h\boldsymbol{J}_h)$	iterations
1	1	2	5	38	1.00	1
2	1/2	8	16	153	1.32	4
3	1/4	32	56	646	1.68	6
4	1/8	128	208	2,650	2.17	6
5	1/16	512	800	10,670	2.34	8
6	1/32	2,048	3,136	42,810	2.40	8
7	1/64	8,192	12,416	–	–	8

Then there exists a constant C independent of h and N such that the eigenvalues of $\boldsymbol{K}_h\boldsymbol{J}_h$ lie in the interval $[1-\delta, 1]$ where $\delta = C/(C+2m)$, m denoting the number of smoothings.

For standard elliptic operators many smoothers can be shown to satisfy the hypotheses, the simplest of which is the scalar smoother. However, the proof for the scalar smoother and some others fails in $\boldsymbol{H}(\mathrm{div})$ and the multigrid preconditioner constructed with these smoothers is not effective. We shall consider an additive Schwarz smoother, defined in the following way. For each vertex of the mesh, consider the patch of elements containing that vertex. These patches form an overlapping covering of Ω and so determine an additive Schwarz operator. We use this operator as our smoother. The verification of the first hypothesis is routine. The standard proof of the second fails, but the difficulty can be surmounted by again using the discrete Helmholtz decomposition in a manner similar to that used for the proof of domain decomposition. The complete proof is given in [Pes72].

6 Numerical Results

First we made a numerical study of the condition number of $\boldsymbol{J}_h$ and the effect of preconditioning. In Table 1.1, the level m mesh is a uniform triangulation of the unit square into 2^{2m-1} triangles and has mesh size $h = 1/2^{m-1}$. The space $\boldsymbol{V}_h$ is taken as the Raviart–Thomas space of index 0 on this mesh. The preconditioner $\boldsymbol{K}_h$ is the V-cycle multigrid preconditioner using one application of the standard additive Schwarz smoother with the scaling factor taken to be 1/2. The fifth column of the table clearly displays the expected growth of the condition number of $\boldsymbol{J}_h$ as $O(h^{-2})$, and the sixth column the boundedness of the condition number of the preconditioned operator $\boldsymbol{K}_h\boldsymbol{J}_h$.

As a second numerical study, we used the Raviart–Thomas mixed method to solve the factored Poisson equation

$$\boldsymbol{u} = \mathbf{grad}\, p, \quad \operatorname{div} \boldsymbol{u} = g \quad \text{in } \Omega, \qquad p = 0 \quad \text{on } \partial\Omega,$$

again on the unit square using the same sequence of meshes as in the first example. We chose $g = 2(x^2+y^2-x-y)$ so that $p = (x^2-x)(y^2-y)$. The discrete solution $(\boldsymbol{u}_h, p_h)$

belongs to the space $\boldsymbol{V}_h \times S_h$, with $\boldsymbol{V}_h$ the Raviart–Thomas space described above and S_h the space of piecewise constant functions on the same mesh. We solved the discrete equations both with a direct solver and by using the minimum residual method preconditioned with the block diagonal preconditioner having as diagonal blocks $\boldsymbol{K}_h$ and the identity (as discussed previously). Full multigrid was used to initialize the minimum residual algorithm. That is, the computed solution at each level was used as an initial guess at the next finer level, beginning with the exact solution on the coarsest (two element) mesh. In Table 1.2, we show the condition number of the discrete operator $\mathcal{A}_h$ and of the preconditioned operator $\mathcal{B}_h\mathcal{A}_h$. While the former quantity grows linearly with h^{-1} (since this is a first order system), the latter remains small.

Table 2 Condition numbers for the indefinite operator $\mathcal{A}_h$ corresponding to the mixed system and for the preconditioned operator $\mathcal{B}_h\mathcal{A}_h$.

level	h	$\dim \boldsymbol{V}_h$	$\dim S_h$	$\kappa(\mathcal{A}_h)$	$\kappa(\mathcal{B}_h\mathcal{A}_h)$
1	1	5	2	8.25	1.04
2	1/2	16	8	15.0	1.32
3	1/4	56	32	29.7	1.68
4	1/8	208	128	59.6	2.18
5	1/16	800	512	119	2.34

Acknowledgement

The first author was supported by NSF grants DMS-9205300 and DMS-9500672 and by the Institute for Mathematics and its Applications. The second author was supported by NSF grant DMS-9403552. The third author was supported by The Norwegian Research Council under grants 100331/431 and STP.29643.

REFERENCES

[AFW97] Arnold D. N., Falk R. S., and Winther R. (1997) Preconditioning in H(div) and applications. *Mathematics of Computation* to appear.

[CLMM94] Cai Z., Lazarov R., Manteuffel T., and McCormick S. (1994) First-order system least squares for second-order partial differential equations: Part I. *SIAM J. Numer. Anal.* 31: 1785–1799.

[DW90] Dryja M. and Widlund O. B. (1990) Towards a unified theory of domain decomposition algorithms for elliptic problems. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Proc. Third Int. Conf. on Domain Decomposition Meths.*, pages 3–21. SIAM, Philadelphia.

[PCL94] Pehlivanov A. I., Carey G. F., and Lazarov R. D. (1994) Least-squares mixed finite elements for second-order elliptic problems. *SIAM J. Numer. Anal.* 31: 1368–1377.

[Xu92] Xu J. (1992) Iterative methods by space decomposition and subspace correction. *SIAM Rev.* 34: 581–613.

3

Scalable Substructuring by Lagrange Multipliers in Theory and Practice

Charbel Farhat and Jan Mandel

1 Introduction

The FETI (Finite Element Tearing and Interconnecting) method is a non-overlapping domain decomposition algorithm for the iterative solution of systems of equations arising from the finite element discretization of self-adjoint elliptic partial differential equations. It is based on using direct solvers in subdomains and enforcing continuity at subdomain interfaces by Lagrange multipliers. The dual problem for the Lagrange multipliers is solved by a preconditioned conjugate gradient (PCG) algorithm. The FETI method was developed in [Far91, FR91, FR92], and discussed in detail in the monograph [FR94]. Unlike other related domain decomposition methods using Lagrange multipliers as unknowns [GW88, Rou90], the FETI method uses the null spaces of the subdomain stiffness matrices (rigid body modes) to construct a small "coarse" problem that is solved in each PCG iteration. It was recognized in [FMR94] and proved mathematically in [MT96] that solving this coarse problem accomplishes a global exchange of information between the subdomains and results in a method which, for elasticity problems, has a condition number that grows only polylogarithmically with the number of elements per subdomain, and is bounded independently of the number of subdomains. For time-dependent problems, one has to solve a linear problem with positive definite subdomain matrices in each time step. The coarse space built from null spaces is lost, resulting in deteriorating convergence with growing number of subdomains. Quasi-optimal convergence properties were retained by introducing an artificial coarse space [FCM95]. For plate bending problems, the condition number was observed to grow fast with the number of elements per subdomain [FMR94]. This was resolved by adding to the coarse space Lagrange multipliers that enforce continuity at the corners [MTF]. A related idea has been employed in the Balancing Domain Decomposition (BDD) method for plates [LMV94], where approximate continuity of the iterates at crosspoints is enforced by adding new basis functions associated with corners to the original coarse space [Man93, MB96].

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

While the underlying ideas of FETI and BDD are in a way dual, FETI is not the BDD method applied to the dual problem. The distinguishing features of both FETI and the BDD method is that they are non-overlapping and work for standard plate and shell finite elements used in everyday engineering practice.

The formulation of the FETI method presented here is based on [MTF], where more details can be found. This formulation covers the original FETI for solids as well as extensions to time-dependent problems and plates and shells. The extension to shells and practical results draw partially on [FCMR95, FM95].

2 Abstract Formulation of FETI

Let Ω be a domain in $\Re^2$ decomposed into N_s non-overlapping subdomains Ω_1, Ω_2, ..., Ω_{N_s}. We assume that there is a conforming finite element discretization defined on Ω, such that each subdomain is a union of some of the elements. The discrete problem arising from this discretization can be formulated as the minimization of the energy subject to intersubdomain continuity conditions,

$$\mathcal{E}(u) = \frac{1}{2}u^T K u - f^T u \rightarrow \min \qquad \text{subject to } Bu = 0. \tag{1}$$

Here,

$$u = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{N_s} \end{bmatrix}, \quad f = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{N_s} \end{bmatrix}, \quad K = \begin{bmatrix} K_1 & 0 & \dots & 0 \\ 0 & K_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & K_{N_s} \end{bmatrix},$$

with u_s, K_s, and f_s being the vector of degrees of freedom, the local stiffness matrix, and the load vector, respectively, associated with the subdomain Ω_s, and $B = [B_1, B_2, \dots, B_{N_s}]$ a given matrix such that $Bu = 0$ expresses the condition that the values of the degrees of freedom associated with two or more subdomains coincide.

The local stiffness matrices K_s and hence K are positive semidefinite. The algorithm will use a given full rank matrix

$$Z = \begin{bmatrix} Z_1 & 0 & \dots & 0 \\ 0 & Z_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & Z_{N_s} \end{bmatrix}, \qquad \operatorname{Range} Z = \operatorname{Ker} K.$$

We assume that the global structure is not floating, that is, the solution of (1) is unique, which is equivalent to $\operatorname{Ker} K \cap \operatorname{Ker} B = \{0\}$.

Introducing Lagrange multipliers λ for the constraint $Bu = 0$, the problem (1) becomes

$$\begin{array}{lclcl} Ku & + & B^T\lambda & = & f \\ Bu & & & = & 0 \end{array} \tag{2}$$

A solution u of the first equation in (2) exists if and only if $f - B^T\lambda \in \operatorname{Range} K$, and

$$u = K^\dagger(f - B^T\lambda) + Z\alpha \ \text{ if } \ f - B^T\lambda \perp \operatorname{Ker} K, \tag{3}$$

where α is to be determined. Substituting u from (3) into the second equation of (2) yields $BK^{\dagger}(f - B^T\lambda) + BZ\alpha = 0$. It follows that λ satisfies the system of equations

$$\begin{aligned} P(F\lambda - d) &= 0, && (4) \\ G^T\lambda &= e, && (5) \end{aligned}$$

where $G = BZ$, $F = BK^{\dagger}B^T$, $d = BK^{\dagger}f$, $P = I - G(G^TG)^{-1}G^T$, $e = Z^Tf$. Note that P is the orthogonal projection onto $\operatorname{Ker} G^T$. It can be proved that $(G^TG)^{-1}$ exists [FR94, MTF].

It is easy to see that any two solutions λ of (4), (5) can differ only by a vector from $\operatorname{Ker} B^T$, and that any solution λ of (4), (5) yields the same solution u of (1) by (3) with $\alpha = -(G^TG)^{-1}G^T(d - F\lambda)$.

The physical interpretation is that the Lagrange multipliers λ are *interface forces and moments*. From (3) and the definition of F, the residual $P(F\lambda - d) = -Bu$ has the interpretation of *jumps in the values of the degrees of freedom* between subdomains. The condition $f - B^T\lambda \perp \operatorname{Ker} K$ means that the action of the loads and intersubdomain forces and moments does not excite rigid body motions.

To obtain more flexibility in the algorithm design, we add to the system (4), (5) a redundant weighted residual condition, and require that all iterates satisfy along with (5) a weighted residual condition

$$C^TP(F\lambda - d) = 0, \qquad (6)$$

where C is another given matrix. The conditions (5), (6) will be enforced throughout the iterations by projecting the increments. For applications to static problems with solid elements, the additional constraint (6) is not necessary, but a proper choice of C is essential for time-dependent problems as well as plate and shell problems.

Increments that preserve (5), (6) form the subspace

$$V' = \{\mu | G^T\mu = 0, C^TPF\mu = 0\}$$

The operator PF is symmetric on $\operatorname{Ker} G^T$ in the sense that

$$\langle PF\lambda, \lambda' \rangle = \langle \lambda, PF\lambda' \rangle, \text{ for all } \lambda, \lambda' \in \operatorname{Ker} G^T,$$

and positive definite on the factorspace $\operatorname{Ker} G^T / \operatorname{Ker} B^T$, cf., [MTF].

To get an initial approximation λ_0 that satisfies (5), (6), we solve a system of equations for a given $\bar{\lambda}_0$

$$\left.\begin{array}{lcrcr} G^TF(\bar{\lambda}_0 + G\alpha + C\beta) & + & G^TG\mu & = & G^Td \\ C^TF(\bar{\lambda}_0 + G\alpha + C\beta) & + & C^TG\mu & = & C^Td \\ G^T(\bar{\lambda}_0 + G\alpha + C\beta) & & & = & e \end{array}\right\} \qquad (7)$$

for unknowns α, β, μ, and set $\lambda_0 = \bar{\lambda}_0 + G\alpha + C\beta$. We will use an analogous process to update a tentative search direction so that it satisfies (6): given $\bar{\lambda}$, one finds a projected search direction $\lambda = \bar{\lambda} + G\alpha + C\beta$, with α, β determined from

$$\begin{array}{lcrcr} G^TF(\bar{\lambda} + G\alpha + C\beta) & + & G^TG\mu & = & 0 \\ C^TF(\bar{\lambda} + G\alpha + C\beta) & + & C^TG\mu & = & 0 \\ G^T(\bar{\lambda} + G\alpha + C\beta) & & & = & 0 \end{array}$$

Then $\lambda = Q\bar{\lambda}$, with Q given by

$$Q = I - \begin{bmatrix} G & C & 0 \end{bmatrix} \begin{bmatrix} G^TFG & G^TFC & G^TG \\ C^TFG & C^TFC & C^TG \\ G^TG & G^TC & 0 \end{bmatrix}^{\dagger} \begin{bmatrix} G^TF \\ C^TF \\ G^T \end{bmatrix},$$

where the superscript † denotes a generalized inverse. It can be proved that [MTF]

$$Q^2 = Q, \qquad \operatorname{Range} Q^T + \operatorname{Ker} B^T = \operatorname{Range} PF + \operatorname{Ker} B^T. \tag{8}$$

Our formulation of the generalized FETI method is now the method of conjugate gradients in the space V' for the operator PF, preconditioned by QDQ^T, where D is symmetric positive semidefinite. It follows from (8) that the preconditioner QDQ^T can be replaced by QD without changing the method. Therefore, the following algorithm is obtained.

Algorithm 1 (Generalized FETI) *Given an initial $\bar{\lambda}_0$, compute the initial λ_0 using (7), and compute the initial residual by*

$$r_0 = P(F\lambda_0 - d).$$

Repeat for $k = 1, 2, \ldots$ until convergence:

$$\begin{aligned} z_{k-1} &= Dr_{k-1} \\ y_{k-1} &= Qz_{k-1} \\ \xi_k &= r_{k-1}^T y_{k-1} \\ p_k &= y_{k-1} + \frac{\xi_k}{\xi_{k-1}} p_{k-1} \qquad (p_1 = y_0) \\ \nu_k &= \frac{\xi_k}{p_k^T PFp_k} \\ \lambda_k &= \lambda_{k-1} + \nu_k p_k \\ r_k &= r_{k-1} - \nu_k PFp_k \end{aligned}$$

3 Selection of Common Algorithm Components

Continuity Constraint $Bu = 0$

For a node x_i at the intersection of two subdomains $\partial\Omega_r \cap \partial\Omega_s$, we define the continuity constraint on the displacement degrees of freedom by

$$(Bw)_{rs}(x_i) = \sigma_{rs}(w_r(x_i) - w_s(x_i)) = 0.$$

We use a similar condition for derivative or rotation degrees of freedom, if present. Here, $\sigma_{rs} = 1$ or $\sigma_{rs} = -1$ is a constant assigned to the edge (in 2D) or side (in 3D). In particular, the entries of B are $-1, 0, +1$, and they are constant along an edge or side between subdomains. Note that this construction of B results in redundant constraints at all degrees of freedom that belong to more than two subdomain. This slightly increases the number of the Lagrange multipliers and complicates the analysis, but makes a simpler parallel implementation possible.

Dirichlet Preconditioner

Decompose the space of all the degrees of freedom into the space of the degrees of freedom lying on the subdomain interfaces, and the degrees of freedom internal to the subdomains

$$W = W_b \times W_i,$$

where the subscript b denotes the block of degrees of freedom on subdomain boundaries, and the subscript i denotes degrees of freedom internal to the subdomains. Then,

$$B = [B_b, 0],$$

since B has nonzero entries for the subdomain interface degrees of freedom only. Also,

$$Z = \begin{bmatrix} Z_b \\ Z_i \end{bmatrix}, \qquad G = BZ = B_b Z_b, \qquad \operatorname{Ker} B^T = \operatorname{Ker} B_b^T.$$

Let S be the Schur complement of K obtained by elimination of the degrees of freedom internal to all subdomains:

$$S = K_{bb} - K_{bi} K_{ii}^{-1} K_{ib}. \tag{9}$$

It is easy to see that

$$F = BK^{\dagger} + B^T = B_b S^{\dagger} B_b^T, \tag{10}$$

and that $\operatorname{Ker} S = \operatorname{Range} Z_b$. It is well known that the evaluation of the matrix-vector product $S^{\dagger} u$ reduces to the solution of independent *Neumann problems* on all subdomains. Analogously to (10), we choose $D = B_b S B_b^T$, giving the preconditioner

$$QD = QB_b S B_b^T. \tag{11}$$

This preconditioner is called the *Dirichlet preconditioner*, since evaluating the matrix-vector product Sr is equivalent to solving independent Dirichlet problems on all subdomains.

Lumped Preconditioner

This is a simplified version of the Dirichlet preconditioner (11), which trades mathematical quasi-optimality for a lower cost per PCG iteration. The Schur complement S of K obtained from (9) is replaced simply by its leading term K_{bb}. This is equivalent to "lumping" each subdomain stiffness on its interface boundary. The resulting preconditioner is given by

$$QD = QB_b K_{bb} B_b^T \tag{12}$$

4 Special Instances of FETI

FETI for Solid Mechanics (Second-Order Elasticity)

The original FETI algorithm [Far91, FR91, FR92] is obtained by omitting the condition (6). Then, Q becomes the identity, and an initial approximation λ_0 is only required to satisfy $G^T\lambda_0 = e$. It was proved in [MT96] that for the Laplace equation, P1 conforming elements, and the Dirichlet preconditioner both in 2D and 3D, and under the usual technical assumptions about the shape regularity of the elements and the subdomains, one has the following upper bound on the condition number

$$\kappa = \frac{\lambda_{max}(QDPF)}{\lambda_{min}(QDPF)} \leq C\left(1 + \log\frac{H}{h}\right)^{\gamma} \tag{13}$$

where h is the characteristic element size, H the characteristic subdomain size, and $\gamma = 3$. If there are no nodes shared between more than two subdomains, and in some other special cases, (13) holds with $\gamma = 2$.

The bound (13) no longer holds for the lumped preconditioner, but one observes a superconvergence effect instead [FMR94]. Because the operator PF is a discretization of the inverse of a differential operator, which is compact, the eigenvalues are clustered around zero. Since the convergence of conjugate gradients after k steps is determined by the spectrum left after removing k extremal eigenvalues, this distribution of eigenvalues results in fast convergence. Unfortunately, as the number of subdomains increases, the spectrum fills in and the superconvergence effect is observed to disappear.

FETI for Time-dependent Problems

The solution of time-dependent problems by an implicit method calls for the repeated solution of linear systems with the subdomain matrices K_s of the form

$$K_s = \tilde{K}_s + (\Delta t)^{-1} M_s, \tag{14}$$

where $\tilde{K}_s$ now denotes the subdomain stiffness matrix, M_s is the subdomain mass matrix and Δt is the time step. Because the mass matrix is positive definite, $\operatorname{Ker} K = \{0\}$, Z is void. Therefore, the natural coarse problem for the unknowns α is lost and the number of iterations increases with the number of subdomains. This can be corrected by the selection $C = B\tilde{Z}$, where $\tilde{Z}$ is chosen so that $\tilde{Z} = \operatorname{diag}\tilde{Z}_s$, $\operatorname{Range}\tilde{Z}_s = \operatorname{Ker}\tilde{K}_s$. Then, it was again observed that the number of iterations is independent on the number of subdomains. It was proved that the iterates approach the static case in the following sense. Consider the FETI iterative process on a linear system with the matrices K_s from (14) with $0 < \Delta t \leq +\infty$, and a fixed right hand side. Let $\lambda^k(\Delta t)$ denote the approximate solution after k iterations of FETI for a given Δt. Then, for all k,

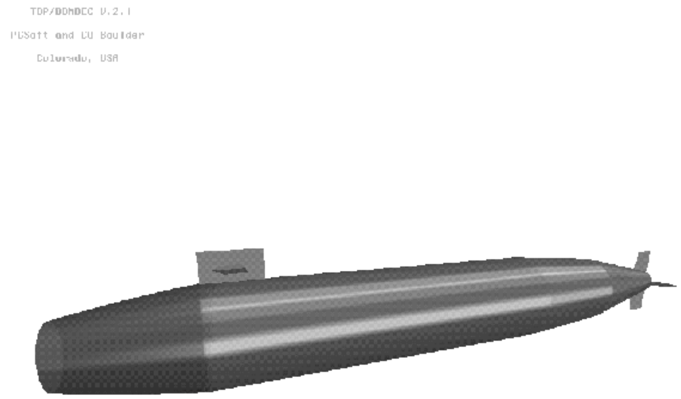
$$\lim_{\Delta t \to +\infty} \lambda^k(\Delta t) = \lambda^k(+\infty).$$

For further details, see [FCM95].

FETI for Plates

Here, the columns of C are chosen as vectors with a one at the position of the Lagrange multiplier that enforces the continuity of the transversal displacement at a crosspoint, and zeroes elsewhere. A crosspoint is an interface node adjacent to at least three subdomains or to two subdomains and the complement of Ω. That is,

Figure 1 The domain splitting for a general operator $\mathcal{A}$ ($n = 3$ & $N = 4$)



Lagrange multipliers that correspond to crosspoints are enforced exactly throughout the iterations.

The condition number bound (13) was proved in [MTF95, MTF] for a general class of plate bending elements that have the property that the local stiffness matrix of the element is spectrally equivalent to that of the HCT element for the biharmonic equation [LMV94]:

$$c_1 K_T^{HCT} \leq K_T \leq c_2 K_T^{HCT} i \tag{15}$$

where K_T^{HCT} is the reduced HCT element stiffness matrix of the biharmonic equation [CT66], with the rotations interpreted as derivatives of the transversal displacement, and K_T is the element stiffness matrix for a triangular or rectangular element with one displacement and two rotation degrees of freedom per node. The spectral equivalence (15) was proved in [LMV] for the particular case of the DKT element [BBH80], and for a general class of non-locking $P1$ Reissner-Mindlin elements that have the element energy functional equivalent to

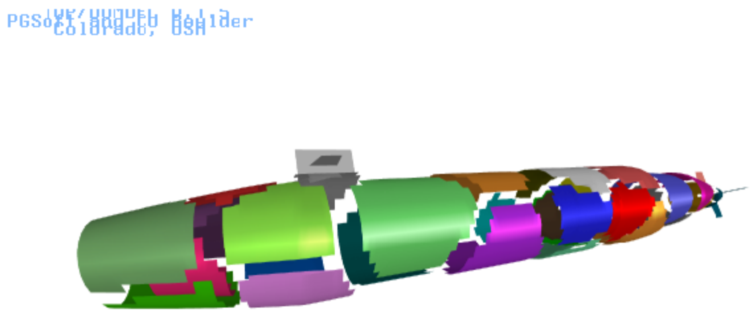
$$\int_T |\nabla \theta|^2 \, dx + \frac{1}{t^2 + h^2} \int_T |\theta - \nabla u|^2 \, dx$$

with $u \in P_1(T), \theta \in (P_1(T))^2$, $h = \text{diam}(T)$, u the transversal displacement, and θ the rotation. This includes the DKT plate bending element as restated in [Pit87].

FETI for Shells

The ideas and theory governing the FETI method for plates [FCMR95, FM95] suggest that, for shell problems, the continuity of the component of the displacement field that is normal to the shell surface should be enforced at the substructure crosspoints throughout the PCG iterations. One approach for implementing this requirement and bypassing the difficulties associated with defining normals for non-smooth shell surfaces consists in enforcing the continuity of the displacement field at the substructure crosspoints in the direction of all three coordinate axes. Clearly,

Figure 2 A 30-substructure mesh partition



this would automatically enforce the continuity of the normal component of the displacement field at the crosspoints, while requiring only a minor modification of the implementation of the FETI method for plates. More precisely, only the construction of the C matrix needs to be modified to have a one at the position of each of the three Lagrange multipliers that enforce the continuity of each of the three displacement degrees of freedom at a crosspoint. In [FCMR95], the authors have shown numerically that, even for irregular shell problems with junctures, such an extension of the FETI method preserves the quasi-optimal convergence properties proved mathematically in [MTF95, MTF] for plate problems.

However, the extension of the FETI method to shell problems summarized above generates a coarse crosspoint problem that is three times larger than that for plate problems, because the continuity of all three displacement degrees of freedom rather than the transversal displacement is enforced at the substructure crosspoints. Hence, wherever the shell structure has a smooth surface, one can enforce only the continuity of the normal component of the displacement field at a crosspoint. This is done by setting $C_{ij} = n_x$, $C_{i+1\ j} = n_y$, $C_{i+2\ j} = n_z$ at that crosspoint and $C_{ij} = 0$ elsewhere, and incurs the same computational cost as for plate problems. Here, n_x, n_y, and n_z denote the three components of the normal to a shell surface at a given crosspoint.

5 Parallel Implementation and Computational Results

The parallel implementation of the FETI method is straightforward, except for the solution of the coarse problem, which has been discussed in detail in [FC94, Far95]. Because of space limitation, we focus here on illustrating only the scalability properties of this method with respect to the number of substructures and processors. The additional scalability of the FETI method with respect to the mesh size has already been demonstrated and reported in all the FETI references cited in this paper.

For this purpose, we consider the stress analysis on a Paragon XP/S system of a submarine structure loaded by a standing pressure wave (Fig. 1). The finite element model contains 60332 nodes, 120064 three-noded shell elements, a total of 361735 active degrees of freedom, and many structural junctures. The mesh is partitioned into 30, 40, 60, and 80 substructures with good aspect ratios [FMB95] for parallel

Table 1 Performance results for a submarine shell structure with 361735 degrees of freedom on a Paragon XP/S parallel processor

# of substructures	# of processors	# of iterations	CPU time coarse problem	Total CPU time
30	30	93	182 sec.	875 sec.
40	40	94	178 sec.	751 sec.
60	60	105	203 sec.	483 sec.
80	80	87	162 sec.	309 sec.

computations on a Paragon XP/S system (2).

Four structural analyses were performed using the FETI method for shells. The corresponding performance results are summarized in Table 1.

Clearly, scalability is well demonstrated for the solution of the coarse problems as well as the solution of the overall problem. The size of the coarse problem increases with the number of substructures and processors, but the CPU time elapsed in forming and solving iteratively the repeated coarse problems is shown to remain almost constant. Moreover, the convergence rate is observed to be almost independent of the number of substructures, and the measured total solution time decreases superlinearly with the number of processors.

Acknowledgement

This research was partially supported by ONR grant N-00014-95-1-0663, and NSF grants ASC-9217394 and ASC-9121431.

REFERENCES

[BBH80] Batoz J. L., Bathe K. J., and Ho W. H. (1980) A study of three-node triangular bending element. *Int. J. Numer. Methods Engrg.* 15: 1771–1812.

[CT66] Clough R. W. and Tocher J. L. (1966) Finite element stiffness matrices for analysis of plate bending. In *Proc. 1965 Conf. Matrix Methods Struct. Mech., Wright-Patterson AFB, Ohio, AFFDL-TR-66-80*, pages 515–546.

[Far91] Farhat C. (1991) Lagrange multiplier based divide and conquer finite element algorithm. *J. Comput. Sys. Engrg.* 2: 149–156.

[Far95] Farhat C. (1995) Optimizing substructuring methods for repeated right hand sides, scalable parallel coarse solvers, and global/local analysis. In Keyes D., Saad Y., and Truhlar D. G. (eds) *Domain-Based Parallelism and Problem Decomposition Methods in Computational Science and Engineering*, pages 141–160. SIAM, Philadelphia.

[FC94] Farhat C. and Chen P. S. (1994) Tailoring domain decomposition methods for efficient parallel coarse grid solution and for systems with many right hand sides. *Contemporary Mathematics* 180: 401–406. Proceedings of the 7th International

Symposium on Domain Decomposition Methods, Penn State, November 1993.

[FCM95] Farhat C., Chen P. S., and Mandel J. (1995) Scalable Lagrange multiplier based domain decomposition method for time-dependent problems. *Int. J. Numer. Meth. Engrg.* 38: 3831–3853.

[FCMR95] Farhat C., Chen P.-S., Mandel J., and Roux F.-X. (1995) The two-level FETI method - Part II: Extension to shell problems, parallel implementation and performance results. Comp. Meth. Appl. Mech. Engrg, to appear.

[FM95] Farhat C. and Mandel J. (1995) The two-level FETI method for static and dynamic plate problems - Part I: An optimal iterative solver for biharmonic systems. Technical Report UCB/CAS Report CU-CAS-95-23, Center for Aerospace Structures, University of Colorado at Boulder. Comp. Meth. Appl. Mech. Engrg, submitted.

[FMB95] Farhat C., Maman N., and Brown G. (1995) Mesh partitioning for implicit computations via iterative domain decomposition: optimization of the subdomain aspect ratio. *Int. J. Numer. Meth. Engrg.* 38: 989–1000.

[FMR94] Farhat C., Mandel J., and Roux F.-X. (1994) Optimal convergence properties of the FETI domain decomposition method. *Comput. Methods Appl. Mech. Engrg.* 115: 367–388.

[FR91] Farhat C. and Roux F.-X. (1991) A method of finite element tearing and interconnecting and its parallel solution algorithm. *Int. J. Numer. Meth. Engng.* 32: 1205–1227.

[FR92] Farhat C. and Roux F.-X. (1992) An unconventional domain decomposition method for an efficient parallel solution of large-scale finite element systems. *SIAM J. Sci. Stat. Comput.* 13: 379–396.

[FR94] Farhat C. and Roux F.-X. (1994) Implicit parallel processing in structural mechanics. *Comput. Mech. Advances* 2: 1–124.

[GW88] Glowinski R. and Wheeler M. F. (1988) Domain decomposition and mixed finite element methods for elliptic problems. In Glowinski R., Golub G. H., Meurant G. A., and Périaux J. (eds) *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 144–172. SIAM, Philadelphia.

[LMV] Le Tallec P., Mandel J., and Vidrascu M.A Neumann-Neumann domain decomposition algorithm for solving plate and shell problems. SIAM J. Numer. Anal., to appear.

[LMV94] Le Tallec P., Mandel J., and Vidrascu M. (1994) Balancing domain decomposition for plates. *Contemporary Mathematics* 180: 515–524. Proceedings of the 7th International Symposium on Domain Decomposition Methods, Penn State, November 1993.

[Man93] Mandel J. (1993) Balancing domain decomposition. *Comm. in Numerical Methods in Engrg.* 9: 233–241.

[MB96] Mandel J. and Brezina M. (1996) Balancing domain decomposition for problems with large jumps in coefficients. *Mathematics of Computation* 65: 1387–1401.

[MT96] Mandel J. and Tezaur R. (1996) Convergence of a substructuring method with Lagrange multipliers. *Numerische Mathematik* 73: 473–487.

[MTF] Mandel J., Tezaur R., and Farhat C.A scalable substructuring method by Lagrange multipliers for plate bending problems. Submitted.

[MTF95] Mandel J., Tezaur R., and Farhat C. (1995) Optimal Lagrange multiplier based domain decomposition method for plate bending problems. UCD/CCM Report 61, Center for Computational Mathematics, University of Colorado at Denver.

[Pit87] Pitkäranta J. (1987) On a simple finite element method for plate bending problems. In Hackbusch W. and Witsch K. (eds) *Numerical Techniques in Continuum Mechanics*, number 16 in Notes on Numerical Fluid Mechanics, pages 84–101. Vieweg, Braunschweig/Wiesbaden. Proceedings of 2nd GAMM-Seminar, Kiel, January 1986.

[Rou90] Roux F.-X. (1990) Acceleration of the outer conjugate gradient by reorthogonalization for a domain decomposition method with Lagrange multiplier. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 314–321. SIAM, Philadelphia.

4

An Asymptotically Optimal Substructuring Method for the Stokes Equation

Boris N. Khoromskij and Gabriel Wittum

1 Introduction

In this paper, we propose and analyze an asymptotically optimal Schur complement interface reduction for the Stokes equation on plane polygonal domains. It is based on using special Poincaré-Steklov (PS) operators, see also [QV91]. We refer to [KW96] for the related results based on a coupling of the stream function-vorticity formulation and the decomposition approach from [GP79]. The multigrid methods of finite elements (FE) for the Stokes and Navier-Stokes equations have been considered, e.g. in [Wit89].

The main ingredient of our method is an appropriate factorization of the matrix-valued traction operator $\mathbf{S}_T^{-1} : \mathbf{u} \rightarrow (\sigma_{nn}, \sigma_{n\tau})^T$ which maps the trace of the velocity vector into the normal and shear stress components σ_{nn} and $\sigma_{n\tau}$. We introduce a symmetric and positive definite (*s.p.d.*) Poincaré-Steklov operator S_{st} for the Stokes equation, see (10), which maps the trace of the pressure into the normal velocity component under the constraints $u_{\tau|\Gamma} = div\mathbf{u}_{|\Gamma} = 0$. This interface operator admits a stable FE approximation providing an asymptotically optimal stiffness matrix compression. We study the mapping properties of the continuous PS operator and briefly discuss the corresponding discrete FE approximations. In the case of a rectangular domain, we apply the algorithm of the complexity $O(N \log^2 N)$ for the fast Schur complement matrix-vector multiplication, where N is the number of degrees of freedom on the (subdomain) boundary, see [KW97]. For domains composed of $M \geq 1$ rectangular substructures, our interface reduction is shown to have a complexity $O(MN \log^{q_r} N)$, where $q_r = 2$ for the multilevel BPX interface preconditioner [JHBX90] and $q_r = 3$ in the case of a BPS type [BPS86] preconditioner. Using an interface reduction by the refined skeleton in the case of polygonal boundaries, see [Kho96, KP95, KS96, KW96], yields an algorithm of the same complexity as above, where $q_r + 1$ must be substituted for q_r. The approach proposed may be extended to the $3D$ case.

Let $\Omega \in R^2$ be a bounded domain with either a smooth or convex polygonal

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

boundary $\Gamma = \cup_{j=1}^{J} \Gamma_j$ composed of linear pieces Γ_j. For given α, $\nu > 0$, $\mathbf{f} \in L^2(\Omega)^2$ and $\mathbf{g} \in \{\mathbf{u} \in H^{1/2}(\Gamma)^2 : (u_n, 1)_{L^2(\Gamma)} = 0\}$, consider the Stokes equation:
Find $(\mathbf{u}, p) \in X \times M$ *such that*

$$\begin{cases} \alpha\mathbf{u} - \nu\Delta\mathbf{u} + \nabla p = \mathbf{f} & in\,\Omega \in R^2 \\ div\mathbf{u} = 0 & in\ \Omega \\ \mathbf{u} = \mathbf{g} & on\ \Gamma\ , \end{cases} \tag{1}$$

where $M = L_0^2(\Omega) = \{p \in L^2(\Omega);\ \int\limits_{\Gamma} p dx = 0\}$, $X = H^1(\Omega)^2$.

For ease of presentation, consider the case $\alpha = 0$. Denote by $\mathbf{n} = (n_1, n_2)^T$ and $\tau = (-n_2, n_1)^T$ the unit outward normal and tangential vectors, respectively. We use the standard notations $X_0 = H_0^1(\Omega)^2$, $V = \{\mathbf{v} \in X : div\mathbf{v} = 0\}$ and $V_0 = V \cap X_0$ and define the continuous bilinear form $a : X \times X \to R$ by

$$a(\mathbf{u}, \mathbf{v}) := 2 \sum_{i,j=1}^{2} \int\limits_{\Omega} \varepsilon_{ij}(\mathbf{u}) : \varepsilon_{ij}(\mathbf{v}) dx, \qquad \varepsilon_{ij} = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}). \tag{2}$$

The V_0-ellipticity of $a(\cdot,\cdot)$, the trace theorem and validity of the LBB *inf-sup* condition

$$\exists \beta > 0 : \qquad \sup_{\mathbf{v} \in X_0} \frac{(q, div\mathbf{v})_{L^2(\Omega)}}{|\mathbf{v}|_{1,\Omega}} \geq \beta \|q\|_{0,\Omega}, \quad \forall q \in M\,, \tag{3}$$

imply *a priori* estimate $\|\mathbf{u}\|_{1,\Omega} + \|p\|_{0,\Omega} \leq c\Big(\|\mathbf{f}\|_{-1,\Omega} + \|\mathbf{g}\|_{1/2,\Gamma}\Big)$, see [GR86, Lad69].

2 Poincaré-Steklov Operators for the Stokes equation

Introduce the Poincaré-Steklov (traction) operator

$$\mathbf{S}_T^{-1}\mathbf{u} \equiv \begin{pmatrix} S_{nn} & S_{\tau n}^T \\ S_{\tau n} & S_{\tau\tau} \end{pmatrix} \begin{pmatrix} u_n \\ u_\tau \end{pmatrix} := \begin{pmatrix} \sigma_{nn} \\ \sigma_{n\tau} \end{pmatrix} : X_{n\tau} \to X'_{n\tau},$$

by the identity

$$(\mathbf{S}_T^{-1}\mathbf{u}, \mathbf{v})_\Gamma := \langle \sigma_{nn}(\mathbf{u}), v_n \rangle_{L^2(\Gamma)} + \langle \sigma_{n\tau}(\mathbf{u}), v_\tau \rangle_{L^2(\Gamma)} = \nu a(\Upsilon\mathbf{u}, \Upsilon\mathbf{v}), \tag{4}$$

$\forall\ \mathbf{u}, \mathbf{v} \in X_{n\tau}(\Gamma)$, where $\Upsilon : X_{n\tau}(\Gamma) \to V$ is the Stokes solution (extension) operator defined by (1) with $\mathbf{f} = 0$. Here, $X_{n\tau}(\Gamma)$ is a trace space of the normal and tangential velocity components

$$X_{n\tau}(\Gamma) := \{\mathbf{v}_{n\tau} = \begin{pmatrix} v_n \\ v_\tau \end{pmatrix} : \mathbf{v} \in H^{1/2}(\Gamma)^2, (v_n, 1)_{L^2(\Gamma)} = 0\}, \quad \|\mathbf{v}_{n\tau}\|_{X_{n\tau}} = \|\mathbf{v}\|_{1/2,\Gamma}.$$

Our purpose is the construction of an efficient FE approximation to the PS operator $\mathbf{S}_T^{-1}$. To that end, we construct such approximations for the inverse to the block-diagonal components S_{nn}^{-1} and $S_{\tau\tau}^{-1}$ defined as the PS operators on the subspaces

$V_n = \{\mathbf{v} \in X_{n\tau} : v_\tau = 0\}$ and $V_\tau := \{v \in X_{n\tau} : v_n = 0\}$ respectively, each of which may be identified with a certain subspace of Y', where

$$Y := \begin{cases} H^{-1/2}(\Gamma) = \left(H^{1/2}(\Gamma)\right)', & \text{if } \Gamma \in C^{1,1} \\ \prod_{j=1}^{J} H^{-1/2}(\Gamma_j) = \left(\prod_{j=1}^{J} \widetilde{H}^{1/2}(\Gamma_j)\right)' & \text{if } \Gamma \text{ is a polygon.} \end{cases}$$

Denote $Y_1' = \{u \in Y' : (u, 1)_{L^2(\Gamma)} = 0\}$.

Consider more precisely the block structure of the 2×2 matrix valued-operator $\mathbf{S}_T^{-1}$. First introduce the basic PS operators associated with the Laplace and biharmonic equations (see [KW96, KS96] for the corresponding variational formulations)

$$S_\Delta^{-1} : \mu \to \frac{\partial}{\partial n} u_{|\Gamma} \in H^{-1/2}(\Gamma); \qquad \begin{cases} \Delta u = 0, \;\; u \in H^1(\Omega) \\ u_{|\Gamma} = \mu \in H^{1/2}(\Gamma), \end{cases} \tag{5}$$

$$S_{\Delta^2}^{-1} : \gamma \to -\Delta\psi_{|\Gamma} \in Y; \qquad \begin{cases} \Delta^2 \psi = 0, \;\; \psi \in H^2(\Omega) \cap H_0^1(\Omega) \\ \frac{\partial \psi}{\partial n}_{|\Gamma} = \gamma \in Y'. \end{cases} \tag{6}$$

Introduce the operator $\widetilde{S}_\Delta : g \to \psi_{|\Gamma}$, where $\psi \in H^1(\Omega)$ solves the equation

$$\begin{cases} \Delta\psi = \frac{1}{mes\,\Omega} \int\limits_\Gamma g ds \;\; \text{in } \Omega \\ \frac{\partial\psi}{\partial n} = g \;\; \text{on } \Gamma. \end{cases} \tag{7}$$

This operator coincides with S_Δ for $g \in H_1^{-1/2}(\Gamma) = \{u \in H^{-1/2}(\Gamma) : (u, 1)_{L^2(\Gamma)} = 0\}$. Let $D = \frac{d}{d\tau}$ and $D^{-1}u = \int\limits_{\tau_0}^{\tau} u(s)ds$, $\forall u \in H_1^{-1/2}(\Gamma)$. Note that the operators S_Δ^{-1} and D provide isomorphisms from $H_1^{1/2}(\Gamma) = \{u \in H^{1/2}(\Gamma) : (u, 1)_{L^2(\Gamma)} = 0\}$ onto $H_1^{-1/2}(\Gamma)$ and $Ker\, S_\Delta^{-1} = Ker\, D = span\{1\}$. The operator $S_\Delta^{-1} : H_1^{1/2}(\Gamma) \to H^{-1/2}(\Gamma)$ is *s.p.d.*, while $D = -D'$ is a skew-symmetric one. Due to [KS96], we know that the mapping $S_{\Delta^2}^{-1} : Y' \to Y$ is continuous and *s.p.d.*

Lemma 2.1 *The operator $\mathbf{S}_T^{-1} : X_{n\tau} \to X_{n\tau}'$ is continuous, symmetric and positive semidefinite. The representation*

$$\mathbf{S}_T^{-1}\mathbf{u} = \begin{pmatrix} -D^{-1} S_\Delta^{-1} S_{\Delta^2}^{-1} \widetilde{S}_\Delta^{-1} D^{-1} & -D^{-1} S_\Delta^{-1} S_{\Delta^2}^{-1} - 2D \\ S_{\Delta^2}^{-1} \widetilde{S}_\Delta^{-1} D^{-1} + 2D & S_{\Delta^2}^{-1} \end{pmatrix} \begin{pmatrix} u_n \\ u_\tau \end{pmatrix} \tag{8}$$

holds for $\forall \mathbf{u} \in X_{n\tau}(\Gamma)$.

Proof. The first assertion follows from definition (4) along the line of the proof of Theorem 3.1. To prove (8), we pass to the stream function-vorticity formulation $\mathbf{u} = \mathbf{curl}\psi$, $\psi \in H^2(\Omega)$, $\psi(x_0) = 0$, $x_0 \in \Gamma$

$$\begin{cases} \nu(\Delta\psi, \Delta\varphi)_{L^2(\Omega)} = \langle \mathbf{f}, \mathbf{curl}\varphi \rangle & \forall \varphi \in H_0^2(\Omega) \\ \psi = \int\limits_{x_0}^{x} g_n ds; \qquad \frac{\partial\psi}{\partial n} = -g_\tau & \text{on } \Gamma \end{cases} \tag{9}$$

using the properties of the biharmonic PS operator $S^{-1}_{\Delta^2}$ studied in [KS96]. More detailed analysis of the representation (8) may be found in [KW96]. □

Since FE discretization to the operators D, D^{-1} and S^{-1}_{Δ} is a rather standard topic, a crucial point in the implementation of the matrix-valued operator (8) is an efficient approximation to the operator $S^{-1}_{\Delta^2}$ associated with the bi-Laplacian. A mixed FE approximation $S_{\Delta^2_h}$ to S_{Δ^2} by $P_1 - P_1$ elements has been developed in [KS96]. It was shown to have the complexity $\mathcal{C}(S_{\Delta^2_h}) = O(N \log^q N)$, where $q = 2$ for a rectangular domain and $q = 3$ in the case of convex polygons. However, the corresponding mixed formulation turns out not to satisfy a uniform LBB condition with respect to the mesh parameter $h > 0$. Thus, an optimal error estimate was not achieved in [KS96].

3 A New Interface Reduction by the Trace of the Pressure

To overcome the above drawback and to develop an approach which may be potentially extended to the 3D problems, we introduce the new Poincaré-Steklov operator associated with the Stokes equation, which admits a stable FE approximation and provides a stiffness matrix compression scheme of the same complexity as for the biharmonic operator $S_{\Delta^2_h}$. Let Ω be either a convex polygon or a domain with a smooth boundary. Introduce the operator $S_{st} : Y \to Y'$ by

$$S_{st} : \lambda \to -(\mathbf{u}_\lambda)_{n_{|\Gamma}}, \quad \text{where} \quad \begin{cases} \Delta p_\lambda = 0, \;\; p_{\lambda|\Gamma} = \lambda \in Y \\ \nu \Delta \mathbf{u}_\lambda - \nabla p_\lambda = 0, \\ div \mathbf{u}_{\lambda|\Gamma} = 0; \quad (\mathbf{u}_\lambda)_{\tau|\Gamma} = 0, \end{cases} \tag{10}$$

which maps the trace of the pressure λ into the normal velocity component $(\mathbf{u}_\lambda)_n$ of the solution to (10) (cf. the decomposition approach developed in [GP79]).

Theorem 3.1 *The operator $S_{st} : Y \to Y'$ is continuous and s.p.d. on Y/R, such that $Ker S_{st} = span\{1\}$, implying*

$$S_{st} = -D\widetilde{S}_\Delta S_{\Delta^2} S_\Delta D \;\; \text{and} \;\; S_{\Delta^2} = -\widetilde{S}^{-1}_\Delta D^{-1} S_{st} D^{-1} S^{-1}_\Delta \quad \text{on } Y. \tag{11}$$

There exists continuous and s.p.d. pseudoinverse $S^{-1}_{st} : Y'_1 \to Y/R$. There holds

$$\mathbf{S}^{-1}_T \mathbf{u} = \begin{pmatrix} S^{-1}_{st} & S^{-1}_{st} D\widetilde{S}_\Delta - 2D \\ -\widetilde{S}_\Delta D S^{-1}_{st} + 2D & -S_\Delta D S^{-1}_{st} D \widetilde{S}_\Delta \end{pmatrix} \begin{pmatrix} u_n \\ u_\tau \end{pmatrix}. \tag{12}$$

Sketch of the proof. To prove the mapping properties of S_{st}, we first note that the constraint $div \mathbf{u}_{|\Gamma} = 0$ implies $div \mathbf{u} = 0$ in Ω for any $\mathbf{u} \in H^2(\Omega)$ satisfying (10). We then apply the basic variational formulation of the second equation in (10) (due to the corresponding Green's formula): $\mathbf{u}_\lambda \in X_\tau$,

$$\nu a(\mathbf{u}_\lambda, \mathbf{v}) - (p, div \mathbf{v}) = -\int_\Gamma \lambda v_n ds \qquad \forall\, \mathbf{v} \in X_\tau = \{\mathbf{z} \in X : z_{\tau|\Gamma} = 0\}, \tag{13}$$

which is valid since the conditions $div \mathbf{u}_{\lambda|\Gamma} = 0$ and $(\mathbf{u}_\lambda)_{\tau|_\Gamma} = 0$ yield the representation

$$\sigma_{nn}(\mathbf{u}_\lambda) = -p_\lambda + 2\nu div \mathbf{u}_{\lambda|\Gamma} = -p_\lambda \qquad \text{on } \Gamma.$$

The symmetry and continuity of S_{st} is derived by the variational equation

$$\langle S_{st}\lambda, \mu\rangle_{L^2(\Gamma)} = \nu a(\mathbf{u}_\lambda, \mathbf{u}_\mu), \quad \forall\, \lambda,\ \mu \in Y. \tag{14}$$

Indeed, due to the trace theorem and Korn's inequality, it follows for $\mathbf{u}_\lambda \in X_\tau$

$$\|S_{st}\lambda\|^2_{Y'} = \|(\mathbf{u}_\lambda)_n\|^2_{Y'} \le c\|\mathbf{u}_{\lambda|\Gamma}\|^2_{H^{1/2}(\Gamma)} \le ca(\mathbf{u}_\lambda, \mathbf{u}_\lambda) = \tag{15}$$

$$= \frac{c}{\nu}\langle S_{st}\lambda, \lambda\rangle_{L^2(\Gamma)} \le \frac{c}{\nu}\|S_{st}\lambda\|_{Y'} \cdot \|\lambda\|_Y.$$

The positive definiteness of S_{st} follows from:
a) the norm equivalence (see [KS96])

$$\|\Lambda\mu\|_{L^2(\Omega)} \cong \|\mu\|_Y \qquad \forall \mu \in Y, \tag{16}$$

where the continuous mapping $\Lambda : Y \to L^2(\Omega)$, such that $\Lambda\mu = \varphi$ denotes a solution operator of the Dirichlet problem for the Laplace equation in a *very weak* form

$$\int_\Omega \varphi \Delta z dx = \langle \mu, \frac{\partial z}{\partial n}\rangle_{L^2(\Gamma)} \qquad \forall z \in H^2(\Omega) \cap H^1_0(\Omega), \quad \mu \in Y;$$

b)*inf-sup* condition (3) for the subspace X_0.
In fact, we use (16), (3), the continuity of $a(\cdot,\cdot)$ and obtain

$$\|\lambda\|_Y \le c\|p_\lambda\|_{0,\Omega} \le \sup_{\mathbf{v}\in X_0} \frac{(p_\lambda, div\mathbf{v})}{|\mathbf{v}|_{1,\Omega}} = \tag{17}$$

$$= \nu \sup_{\mathbf{v}\in X_0} \frac{a(\mathbf{u}_\lambda, \mathbf{v})}{|\mathbf{v}|_{1,\Omega}} \le c\nu a(\mathbf{u}_\lambda, \mathbf{u}_\lambda)^{1/2} \le c\nu^{1/2}\langle S_{st}\lambda, \lambda\rangle^{1/2}_{L^2(\Gamma)}.$$

The representations (11) and (12) follow from (8) and from the equivalence between (19) and (7), see also the proof of Theorem 3.2. □

The operator S_{st} provides an alternative representation (12) to the matrix-valued PS operator $\mathbf{S}_T^{-1}$. In this case, we may avoid the stream function-vorticity formulation and construct a stable FE approximation to S_{st}. Moreover, the representation (12) involves only the operators in the normal-tangential (i.e., dimensionally invariant) form and provides a natural base for an extension of the underlying techniques to the $3D$ case. The operator S_{st} also provides an efficient boundary reduction to the Stokes equation (if $\mathbf{f} = 0$) with respect to the trace of the pressure.

Theorem 3.2 *The trace $\lambda = p_{|\Gamma}$ of the solution $(\mathbf{u}, p)$ to the Dirichlet problem (1) (with $\mathbf{f} = 0$) satisfies*

$$\lambda \in Y/R : \quad \langle S_{st}^{-1}\lambda, \mu\rangle_{L^2(\Gamma)} = \langle g_n - (\mathbf{u}_0)_n, \mu\rangle_{L^2(\Gamma)} \qquad \forall \mu \in Y/R, \tag{18}$$

where $\mathbf{u}_0$ solves the following mixed problem for the vector Laplace equation

$$-\nu\Delta\mathbf{u}_0 = 0 \qquad in\ \Omega; \qquad (\mathbf{u}_0)_\tau = g_\tau, \quad div\mathbf{u}_0 = 0\ \ on\ \Gamma. \tag{19}$$

Proof. The unique solvability of (19) is checked by using the substitution $\mathbf{u}_0 = \mathbf{curl}\,\psi, \psi \in H^2(\Omega), \psi(x_0) = 0$, where ψ satisfies (7) such that $\frac{\partial\psi}{\partial n} = g_\tau$ and $\frac{\partial\psi}{\partial\tau} = -(\mathbf{u}_0)_n$. Then the assertion follows from Theorem 3.1. □

Remark 3.1 *Equation (13) has an equivalent form*

$$\nu d(\mathbf{u}_\lambda, \mathbf{v}) + (\nabla p, \mathbf{v})_{L^2(\Omega)} = 0 \qquad \forall \mathbf{v} \in X_\tau, \tag{20}$$

where the bilinear form $d : X \times X \to R$ is defined by

$$d(\mathbf{u}, \mathbf{v}) := (curl\mathbf{u},\, curl\mathbf{v})_{L^2(\Omega)} + (div\mathbf{u},\, div\mathbf{v})_{L^2(\Omega)}. \tag{21}$$

Here, the operator $curl : X \to R$ is given by $curl\mathbf{v} = \frac{\partial v_2}{\partial x_1} - \frac{\partial v_1}{\partial x_2}$. For technical reasons, we further construct a discrete scheme on the base of above defined form $d(\cdot, \cdot)$.

4 A Stable FE Approximation to the Interface Operator

Let Ω be a rectangular domain. Assume $M_h \in M$ and $X_{\tau h} \in X_\tau$ to be the spaces of $P_1\, iso\, P_2/P_1$ FEs, see [Pir89], defined on the regular hierarchical triangulations $\mathcal{T}_h$ and $\mathcal{T}_{h/2}$ of $\overline{\Omega}$. Let $\mathbf{u}_{0h}$ be the discrete solution of (19) based on the FE approximation of the Poisson equation (7) with respect to M_h. Introduce the equations
Given $\lambda_h \in Y_h := M_{h|\Gamma}$, find $p_{\lambda h} \in M_h$, such that $p_{\lambda h} = \lambda_h$ on Γ and

$$(\nabla p_{\lambda h}, \nabla q_h)_{L^2(\Omega)} = 0 \qquad \forall q_h \in M_h \cap H_0^1(\Omega)\ ; \tag{22}$$

Find $\mathbf{u}_{\lambda h} \in X_{\tau h}$, such that:

$$\nu d(\mathbf{u}_{\lambda h}, \mathbf{v}_h) - (p_{\lambda h}, div\mathbf{v}_h) = -(\lambda_h, (\mathbf{v}_h)_n)_{L^2(\Gamma)} \qquad \forall \mathbf{v}_h \in X_{\tau h}\ . \tag{23}$$

For any $\lambda_h \in Y_h$, define FE approximation P_h to S_{st} by

$$\langle P_h\lambda_h, \mu_h\rangle = \nu\, d(\mathbf{u}_{\lambda h}, \mathbf{u}_{\mu h}), \qquad \forall \mu_h \in Y_h\ . \tag{24}$$

The *s.p.d.* operator P_h admits a fast matrix-vector multiplication. The discrete system related to (18) can now be written as a boundary equation with respect to the trace of the pressure

$$\langle P_h\lambda_h, \mu_h\rangle = -((\mathbf{g} - \mathbf{u}_{0h}) \cdot \mathbf{n}\ , \mu_h)_{L^2(\Gamma)} \qquad \forall \mu_h \in Y_h\ . \tag{25}$$

With λ_h satisfying (25), the approximate velocity $\mathbf{u}_h$ and the pressure p_h are given by $\mathbf{u}_h = \mathbf{u}_{0h} + \mathbf{u}_{\lambda h}, \quad p_h = p_{\lambda h}$. Assuming $\mathcal{T}_h$ to be the uniform triangulation, we may prove the main result.

Theorem 4.1 *The operator $P_h : Y_h \to Y_h'$ is s.p.d. on Y_h/R providing the norm equivalence*

$$\nu\,\langle P_h\mu_h, \mu_h\rangle \simeq \|\mu_h\|^2_{Y/R} \qquad \forall \mu_h \in Y_h \tag{26}$$

with constants of equivalence not depending on h. There holds

$$\|\lambda - \lambda_h\|_Y \leq c\, h\, (|\mathbf{u}|_{2,\Omega} + |p|_{1,\Omega})\ . \tag{27}$$

Sketch of the proof. Applying the trace theorem and Korn's inequality, we obtain

$$\langle P_h\lambda_h, \lambda_h\rangle = \nu\, d(\mathbf{u}_{\lambda h}, \mathbf{u}_{\lambda h}) \le c\,|\lambda_h|_{Y/R}\,\|(\mathbf{u}_{\lambda h})_n\|_{H^{1/2}(\Gamma)} \le |\lambda_h|_{Y/R}\, d(\mathbf{u}_{\lambda h}, \mathbf{u}_{\lambda h})^{1/2}.$$

The other direction follows from the norm equivalence $\|p_{\mu h}\|_{0,\Omega} \simeq \|\mu_h\|_Y$, $\forall \mu_h \in Y_h$, see [KS96], the discrete inf-sup condition and continuity of $d(\cdot,\cdot)$. Indeed,

$$p_{\lambda h} \in M_h: \qquad \|\lambda_h\|_{Y/R} \le c\,\|p_{\lambda h}\|_{0,\Omega} \le c \sup_{\mathbf{v}_h \in X_{0h}} \frac{(p_{\lambda h}, div\mathbf{v}_h)}{|\mathbf{v}_h|_{1,\Omega}} =$$

$$= \nu\, c \sup_{\mathbf{v}_h \in X_{0h}} \frac{d(\mathbf{u}_{\lambda h}, \mathbf{v}_h)}{|\mathbf{v}_h|_{1,\Omega}} \le \nu\, c\, d(\mathbf{u}_{\lambda h}, \mathbf{u}_{\lambda h})^{1/2} = \nu^{1/2}\, c\,\langle P_h\lambda_h, \lambda_h\rangle^{1/2}\ .$$

Now (27) follows from (26) and standard error estimates for (22) and (23), see [KW97] for more details. □

Finally, the symmetric and positive definite FE approximation to $\mathbf{S}_T^{-1}$ from (12) is obtained by a substitution of D_h, $\widetilde{S}_{\Delta_h}$ and P_h into (12) instead of the corresponding continuous operators.

Remark 4.1 *Using the discrete operator P_h, we immediately obtain an s.p.d. FE approximation to the biharmonic Poincaré-Steklov operator S_{Δ^2} by $S_{\Delta^2_h} = -S^{-1}_{\Delta_h} D_h^{-1} P_h D_h^{-1} \widetilde{S}^{-1}_{\Delta_h}$ yielding an optimal approximation error and efficient matrix compression. This means that our interface reduction for the Stokes equation provides an efficient solver for the stream function-vorticity formulation as well.*

5 An Interface Reduction by the Domain Decomposition

We consider the *s.p.d.* approximation of A_{Γ_0} by using the operator P_h. To fix the idea, we assume $\overline{\Omega} = \cup_i \overline{\Omega}_i$ to be composed of rectangular subdomains Ω_i. First derive an interface reduction to the equation (1) with the given right-hand side $\mathbf{f} \ne 0$ and $\mathbf{g} = 0$. For any subdomain Ω_i, assume the traction vector $\psi_{0i} = \binom{\sigma_{nn}(\mathbf{u}_{0i})}{\sigma_{n\tau}(\mathbf{u}_{0i})}_{|\Gamma_i}$ of the corresponding particular solution $\mathbf{u}_{0i} \in H^1_0(\Omega_i)^2$ to be given. Define the related trace space on the skeleton $\Gamma_0 = \cup_i \Gamma_i$ by

$$Y_{\Gamma_0} := \{\mathbf{u} = \mathbf{v}_{|\Gamma_0} : \mathbf{v} \in H^1_0(\Omega)^2,\ ((\mathbf{v}_i)_n, 1)_{\Gamma_i} = 0,\ i = 1, ..., M\} \tag{28}$$

and equip it with the norm $\|\mathbf{u}\|_{Y_{\Gamma_0}} = \inf_{\mathbf{z} \in V_0; \mathbf{z}_{|\Gamma_0} = \mathbf{u}} \|\mathbf{z}\|_{H^1(\Omega)}$. The interface reduction to (1) takes the form:

Find $\mathbf{u} \in Y_{\Gamma_0}$, such that $\mathbf{u} = \overline{\mathbf{u}}_{|\Gamma_0}$ ($\overline{\mathbf{u}}$ solves (1)) and satisfies

$$\langle A_{\Gamma_0}\mathbf{u}, \mathbf{v}\rangle_{\Gamma_0} := \sum_{i=1}^{M} (S^{-1}_{iT}\mathbf{u}_i, \mathbf{v}_i)_{\Gamma_i} = \sum_{i=1}^{M} (\psi_{0i}, \mathbf{v}_i)_{\Gamma_i} \quad \forall \mathbf{v} \in Y_{\Gamma_0}. \tag{29}$$

Due to V_0-ellipticity of $a(\cdot,\cdot)$, the continuous and symmetric operator $A_{\Gamma_0} : Y_{\Gamma_0} \to Y'_{\Gamma_0}$ is also positive definite. We approximate S^{-1}_{iT} given by (12) using the *s.p.d.* operator P_h. To avoid the divergence-free constraints $\big((\mathbf{u}_i)_n, 1\big)_{\Gamma_i} = 0$, $i = 1, \dots M$ and then to apply the standard preconditioning techniques, we first extend the interface operator A_{Γ_0} to the constraints-free trace space $\widetilde{Y}_{\Gamma_0} := \{\mathbf{u} = \mathbf{v}_{|\Gamma_0} : \mathbf{v} \in H_0^1(\Omega)^2\}$ preserving the symmetry and the norm equivalence on Y_{Γ_0}. This extension is based on a scaling of the trace of the pressure on any subdomain boundary Γ_i (by an appropriate choice of the constants $p_i = (p, 1)_{L^2(\Gamma_i)}$) and on using of a special coarse mesh space Y_1 responsible for the divergence-free constraints on Γ_i.

Let $Y_1 = span\{\mathbf{g}^i\}_{i=1}^M \subset \widetilde{Y}_{\Gamma_0}$ be the coarse mesh space of the dimension $dim Y_1 = M$ (in general $\Gamma_i \subset supp\, \mathbf{g}^i$), where the normalized basis functions $\mathbf{g}^i$ and the corresponding Gram matrix $\mathcal{G}$ satisfy

$$det\, \mathcal{G} \neq 0, \quad \mathcal{G} = \{g_{ij}\}_{i,j=1}^M, \quad g_{ij} = (\mathbf{g}^i, \mathbf{1}^j)_{\Gamma_0}, \quad \mathbf{1}^i = (1,0)^T \text{ on } \Gamma_i. \tag{30}$$

Then the following splitting into the direct sum $\widetilde{Y}_{\Gamma_0} = Y_{\Gamma_0} \oplus Y_1$ holds, such that $Y_1' = span\{\mathbf{1}^i\}_{i=1}^M = Y_{\Gamma_0}^{\perp}$. Let $\mathbf{S}_\Delta^{-1} : \widetilde{Y}_{\Gamma_0} \to \widetilde{Y}_{\Gamma_0}'$ be the Poincaré-Steklov operator corresponding to the weighted vector Laplacian. Define the operator $A_1 : Y_1 \to Y_1'$ on Y_1 (by an inexact h- harmonic extension of $\mathbf{g}^i$) providing the norm equivalence

$$\langle A_1 \mathbf{g}, \mathbf{g}\rangle_{\Gamma_0} \cong \langle \mathbf{S}_\Delta^{-1} \mathbf{g}, \mathbf{g}\rangle_{\Gamma_0} \qquad \forall\, \mathbf{g} \in Y_1. \tag{31}$$

We then obtain $\langle A_{\Gamma_0} \mathbf{u}, \mathbf{g}\rangle_{\Gamma_0} = 0$ and $\langle A_1 \mathbf{g}, \mathbf{u}\rangle_{\Gamma_0} = 0$ $\forall \mathbf{u} \in Y_{\Gamma_0}$, $\mathbf{g} \in Y_1$ by an appropriate scaling of $p_{|\Gamma_i}$ and by the definition, respectively. The desired extension $\widetilde{A}_{\Gamma_0}$ is now defined for any $\mathbf{u}, \mathbf{v} \in Y_{\Gamma_0}$ and $\mathbf{g}_u, \mathbf{g}_v \in Y_1$ by

$$\langle \widetilde{A}_{\Gamma_0}(\mathbf{u} + \mathbf{g}_u), \mathbf{v} + \mathbf{g}_v\rangle_{\Gamma_0} = \langle A_{\Gamma_0} \mathbf{u}, \mathbf{v}\rangle_{\Gamma_0} + \langle A_1 \mathbf{g}_u, \mathbf{g}_v\rangle_{\Gamma_0}. \tag{32}$$

If we assume the right-hand sides ψ_{0i} to satisfy the compatibility conditions $(\psi_{0i}, \mathbf{g}^i)_{\Gamma_i} = 0$ (by an appropriate scaling of $\sigma_{nn}(\mathbf{u}_{0i})$), then (29) becomes equivalent to the equation

$$\mathbf{u} \in Y_{\Gamma_0} : \quad \langle \widetilde{A}_{\Gamma_0}(\mathbf{u} + \mathbf{g}_u), \mathbf{v}\rangle_{\Gamma_0} = \sum_{i=1}^M (\psi_{0i}, \mathbf{v}_i)_{\Gamma_i} \qquad \forall \mathbf{v} \in \widetilde{Y}_{\Gamma_0} \tag{33}$$

posed on the constraints-free trace space $\widetilde{Y}_{\Gamma_0}$ and providing $\mathbf{g}_u = 0$. Clearly, the operator $\widetilde{A}_{\Gamma_0}$ is symmetric and positive definite. It may be shown to be spectrally equivalent to $\mathbf{S}_\Delta^{-1}$. Thus, one may apply any standard preconditioners (which remain verbatim for the piecewise Laplacian) to solve the equation (33). In particular, the BPS, balancing type and multilevel BPX preconditioners may be constructed for the iterative solving of the interface equation (33). More detailed analysis of the abovementioned preconditioning techniques (also in the presence of right triangular subsructures) may be found in [KW97]. An efficient computation of the residual for the equations (29) and (33) is based on a fast matrix-vector multiplication for the local Schur complement matrices $\mathcal{S}_{iT}^{-1}$ associated with S_{iT}^{-1}. In the case of rectangular domains, the corresponding matrix compression scheme of the complexity $O(N \log^2 N)$ was presented in [KW97]. Here N denotes the number of degrees of freedom on the subdomain boundary. With such compression algorithm, we arrive at the estimate $O(MN \log^{q_r} N)$ for the overall computational complexity of the PCG methods applied to the system (33). Here, $q_r = 2$ for the multilevel BPX preconditioner on the interface and $q_r = 3$ in the case of the BPS preconditioner.

Acknowledgement

The authors wish to thank Professor A. H. Schatz and Professor J. Xu for stimulating discussions and valuable comments.

REFERENCES

[GP79] Glowinski R. and Pironneau O. (1979) On mixed finite element approximation of the stokes problem i. convergence of the approximate solution. *Numer. Math.* 33: 397–424.

[GR86] Girault V. and Raviart P.-A. (1986) *Finite element methods for Navier-Stokes equations.* Springer-Verlag, Berlin.

[JHBS86] J. H. Bramble J. E. P. and Schatz A. H. (1986) The construction of preconditioners for elliptic problems by substructuring. *Math. Comp.* 47: 103–134.

[JHBX90] J. H. Bramble J. E. P. and Xu J. (1990) Parallel multilevel preconditioners. *Math. Comp.* 55: 1–22.

[Kho96] Khoromskij B. N. (1996) On fast computations with the inverse to harmonic potential operators via domain decomposition. *Numer. Linear Algebra with Applications* 3: 91–111.

[KP95] Khoromskij B. N. and Prössdorf S. (1995) Multilevel preconditioning on the refined interface and optomal boundary solvers for the laplace equation. *Advances in Comp. Math.* 4: 331–355.

[KS96] Khoromskij B. N. and Schmidt G. (1996) Asymptotically optimal interface solver for the biharmonic dirichlet problem on convex polygonal domains. *ZAMM* 76, Suppl. 1: 231–234.

[KW96] Khoromskij B. N. and Wittum G. (1996) An asymptotically optimal schur complement reduction for the stokes equation. Technical report, ICA III, Stuttgart University.

[KW97] Khoromskij B. N. and Wittum G. (1997) Sparse interface reduction for two-dimensional stokes equation. Technical report, ICA III, Stuttgart University.

[Lad69] Ladyzhenskaya O. A. (1969) *The mathematical theory of viscous incompressible flow.* Gordon and Breach, New York.

[Pir89] Pironneau O. (1989) *Finite Element Methods for Fluids.* J. Wiley & Sons Ltd., Masson, Paris.

[QV91] Quarteroni A. and Valli A. (1991) Theory and application of steklov-poincaré operators for boundary-value problems. In *Applied and Industrial Mathematics*, pages 179–203. Kluwer AP.

[Wit89] Wittum G. (1989) Multigrid methods for stokes and navier-stokes equation. *Numer. Math.* 54: 543–563.

5

Non-overlapping Domain Decomposition Preconditioners with Inexact Solves

James Bramble, Joseph Pasciak and Apostol Vassilev

1 Introduction

In this paper, we consider the solution of the discrete systems of equations which result from finite element or finite difference approximation of second order elliptic and parabolic boundary problems. To effectively take advantage of modern parallel computing environments, algorithms must involve a large number of tasks which can be executed concurrently. Domain decomposition preconditioning techniques represent a very effective way of developing such algorithms. The parallelizable tasks are associated with subdomain solves.

There are two basic approaches to the development of domain decomposition preconditioners. The first is the so-called non-overlapping approach and is characterized by the need to solve subproblems on disjoint subdomains. Early work was applicable to domains partitioned into subdomains without internal cross-points [BW86], [BPS86b], [Dry89]. To handle the case of cross-points, Bramble, Pasciak and Schatz introduced in [BPS86a] algorithms involving a coarse grid problem and provided analytic techniques for estimating the conditioning of the domain decomposition boundary preconditioner, a central issue in the subject. Various extensions of these ideas were provided in [Wid88] including a Neumann-Dirichlet checkerboard like preconditioner. Subsequently, these techniques were extended to problems in three dimensions in [BPS89] and [Dry88]. A critical ingredient in the three dimensional algorithms was a coarse grid problem involving the solution averages developed in [BPS87]. Related work is contained in [CMW95], [Nep91], [Smi90].

The papers [BPS86b], [BPS86a], [BPS87], [BPS88], and [BPS89] developed domain decomposition preconditioners for the original discrete system. The alternative approach, to reduce to an iteration involving only the unknowns on the boundary, was taken in [BW86], [BPX91], [CMW95], and [Smi90]. The difference in the two techniques is important in that for the first, it is at least feasible to consider replacing the subproblem solves by preconditioners.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

The second approach for developing domain decomposition preconditioners involves the solution of subproblems on overlapping subdomains. For such methods it is always possible to replace the subproblem solution with a preconditioning evaluation [BPWX91]. However, in parallel implementations, the amount of inter-processor communication is proportional to the amount of overlap. These methods loose some efficiency as the overlap becomes smaller [DW94]. Theoretically, they are much worse in the case when there are jumps in coefficients (see, Remark 3.3 below). In contrast, the convergence estimates for correctly designed non-overlapping domain decomposition algorithms are the same as those for smooth coefficients as long as the jumps align with subdomain boundaries.

Thus, it is natural to investigate the effect of inexact solves on non-overlapping domain decomposition algorithms. Early computational results showing that inexact non-overlapping algorithms can perform well were reported in [GW87]. References to other experimental work can be found in [DSW94]. Analysis and numerical experiments with inexact algorithms of Neumann–Dirichlet and Dirichlet types, under the additional assumption of high accuracy of the inexact solves, were given in [B8̈9] and [HLM91]. Their analysis suggests that the inexact preconditioners do not, in general, preserve the asymptotic condition number behavior of the corresponding exact method, even when the forms providing the inexact interior solves are uniformly equivalent to the original.

In this paper, we develop new non-overlapping domain decomposition preconditioners with inexact solves. We provide variations of the exact algorithm considered in [BPS87]. We develop algorithms based only on the assumption that the interior solves are provided by uniform preconditioning forms. The inexact methods exhibit the same asymptotic condition number growth as the one in [BPS87] and are much more efficient computationally. Our algorithms are alternatives to and in many applications less restrictive than the preconditioners in [B8̈9] and [HLM91]. The convergence estimates developed here are independent of jumps of the operator coefficients across subdomain boundaries.

An important aspect of the analysis provided in this paper is that the non-overlapping preconditioners are shown to be of additive Schwarz type. Even though the new methods are inspired by and implemented according to the classical non-overlapping methodology, they can be reformulated as additive Schwarz algorithms with appropriately chosen subspace decompositions.

The first algorithm of this paper involves a coarse subspace utilizing a simple extension defined in terms of the the average value of the function on the boundary. After preparing this manuscript, it has come to our attention that this extension was also used in a recent paper by Bjørstad, Dryja and Vainikko [BDV96] which was presented in the Eight Domain Decomposition meeting in the summer of 1995. Both the present paper and the one just mentioned rely on the use and analysis of a boundary form defined in terms of boundary averages. This boundary form was also analyzed in [BPS87].

The second algorithm in the present paper is a classical domain decomposition algorithm with inexact solves. It is shown to be an additive Schwarz procedure with special subspace decomposition. The particular decomposition depends on the inexact solve and thus needs to be investigated differently from the standard additive Schwarz approach. Finally, the results and analysis of the current paper were presented by the second author at the Seventh Copper Mountain Multigrid Conference in April of 1995.

2 Preliminaries and Notation

In this section we formulate a model elliptic problem and introduce the corresponding finite element discretization. We also outline the guiding principles in constructing our preconditioner.

We consider the Dirichlet problem

$$\mathcal{L}u = f \qquad \text{in} \quad \Omega, \tag{1a}$$

$$u = 0 \qquad \text{on} \quad \partial\Omega, \tag{1b}$$

where f is a given function, $\Omega \subset \mathbb{R}^n$ $(n = 1,\ 2,\ 3)$ is a bounded polyhedral domain with Lipshitz boundary, and

$$\mathcal{L}v = -\sum_{i,j=1}^{n} \frac{\partial}{\partial x_i}\left(a_{ij}\frac{\partial v}{\partial x_j}\right). \tag{2}$$

Here the $n \times n$ coefficient matrix $\{a_{ij}\}$ is symmetric, uniformly positive definite, and bounded above on Ω. This is a classical model problem for a second order uniformly elliptic equation.

The generalized Dirichlet form on Ω is given by

$$\mathcal{A}(v, w) = \sum_{i,j=1}^{n} \int_{\Omega} a_{ij}\partial_i v\, \partial_j w \ dx. \tag{3}$$

This symmetric bilinear form is well defined for functions v and w in the Sobolev space $H^1(\Omega)$. The $L^2(\Omega)$–inner product and the related norm are defined by

$$(v, w)_{\Omega} = \int_{\Omega} vw \ dx$$

and

$$\|v\|_{\Omega}^2 = (v, v)_{\Omega}.$$

Let $H_0^1(\Omega)$ be the Sobolev space obtained by the completion of smooth functions with support in Ω with respect to the norm in $H^1(\Omega)$. The weak formulation of (1) in $H_0^1(\Omega)$ is then given by the following.

Find $u \in H_0^1(\Omega)$ such that

$$\mathcal{A}(u, \varphi) = (f, \varphi), \quad \text{for all} \quad \varphi \in H_0^1(\Omega). \tag{4}$$

Given a finite dimensional subspace $S_h^0(\Omega)$ of $H_0^1(\Omega)$, the standard Galerkin approximation to (4) is defined by:

Find $u_h \in S_h^0(\Omega)$ such that

$$\mathcal{A}(u_h, \varphi) = (f, \varphi), \quad \text{for all} \quad \varphi \in S_h^0(\Omega). \tag{5}$$

To define $S_h^0(\Omega)$, we partition Ω into triangles $\{\tau_i^h\}$ (or tetrahedra) in the usual way. Here h is the mesh parameter and is defined to be the maximal diameter of all such triangles. By definition, these triangles are closed sets. We assume that the triangulation is quasi-uniform. The collection of simplex vertices will be denoted by $\{x_i\}$.

By convention, any union of elements τ_j^h in a given triangulation will be called a mesh subdomain. In the sequel Ω is assumed partitioned into n_d mesh subdomains $\{\Omega_k\}_{k=1}^{n_d}$ of diameter less than or equal to d. The notation Ω_k will be used for the set of all points of a subdomain including the boundary $\partial\Omega_k$.

We now define the finite element spaces. Let $S_h^0(\Omega)$ be the space of continuous piecewise linear (with respect to the triangulation) functions that vanish on $\partial\Omega$. Correspondingly, $S_h^0(\Omega_k)$ will be the space of functions whose supports are contained in Ω_k and hence each function in $S_h^0(\Omega_k)$ vanishes on $\partial\Omega_k$. $S_h(\Omega_k)$ will consist of restrictions to Ω_k of functions in $S_h^0(\Omega)$. Let Γ denote $\bigcup_k \partial\Omega_k$ and let $S_h(\Gamma)$ and $S_h(\partial\Omega_k)$ be the spaces of functions that are restrictions to Γ and $\partial\Omega_k$, respectively, of functions in $S_h^0(\Omega)$. We consider piecewise linear functions for convenience since the results and algorithms to be developed extend to higher order elements without difficulty.

The following additional notation will be used. Let the $L^2(\partial\Omega_k)$-inner product be denoted by

$$\langle u, v\rangle_{\partial\Omega_k} = \int_{\partial\Omega_k} uv \; ds$$

and the corresponding norm by

$$|v|_{\partial\Omega_k} = \langle v, v\rangle_{\partial\Omega_k}^{1/2}.$$

On $S_h(\partial\Omega_k)$, the discrete inner product and norm are defined by

$$\langle u, v\rangle_{\partial\Omega_k,h} = h^{n-1} \sum_{x_i\in\partial\Omega_k} u(x_i)v(x_i)$$

and

$$|v|_{\partial\Omega_k,h} = \langle v, v\rangle_{\partial\Omega_k,h}^{1/2}.$$

Because of the mesh quasi-uniformity, the norm equivalence

$$c\,|v|_{\partial\Omega_k}^2 \le |v|_{\partial\Omega_k,h}^2 \le C\,|v|_{\partial\Omega_k}^2 \tag{6}$$

holds for function $v \in S_h(\partial\Omega_k)$.

Here and in the remainder of the paper, we shall use c and C to denote generic positive constants independent of discretization parameters such as h, d, and subdomain index k. The actual values of these constants will not necessarily be the same in any two instances.

Finally, $\mathcal{D}_k(\cdot,\cdot)$ denotes the Dirichlet inner product on Ω_k defined by

$$\mathcal{D}_k(v, w) = \sum_{i=1}^{n} \int_{\Omega_k} \partial_i v \partial_i w \; dx, \quad \text{for all} \quad v,\, w \,\in\, H^1(\Omega_k). \tag{7}$$

The development of a method for efficient iterative solution of (5) is the subject of our considerations in this section. In particular, using the decomposition of Ω described above, we shall define a bilinear form $\mathcal{B}(\cdot,\cdot)$ on $S_h^0(\Omega) \times S_h^0(\Omega)$ which satisfies the following two basic requirements. First, the solution $W \in S_h^0(\Omega)$ of

$$\mathcal{B}(W,\varphi) = (g,\varphi)_\Omega \quad \text{for all} \quad \varphi \in S_h^0(\Omega), \tag{8}$$

with g given, should be more efficient to compute than the solution of (5). Second, the two forms should be equivalent in the sense that

$$\lambda_1 \mathcal{B}(V,V) \leq \mathcal{A}(V,V) \leq \lambda_2 \mathcal{B}(V,V) \quad \text{for all} \quad V \in S_h^0(\Omega), \tag{9}$$

for some positive constants λ_1 and λ_2 with λ_2/λ_1 not too large. These conditions, though somewhat vague, serve as guidelines for our construction.

3 The Preconditioner $\mathcal{B}(\cdot,\cdot)$

To define our domain decomposition preconditioner, we will need boundary extension operators. For each k, let us define linear extension operators $\mathcal{E}_k : S_h(\partial\Omega_k) \to S_h(\Omega_k)$ by

$$\mathcal{E}_k\phi(x_i) = \begin{cases} \phi(x_i) & \text{for } x_i \in \partial\Omega_k, \\ 0 & \text{for } x_i \in \Omega_k \setminus \partial\Omega_k. \end{cases}$$

We remind that the functions in the finite element spaces defined above are fully determined by their values at the grid nodes and thus it is sufficient to define the extensions $\mathcal{E}_k$ at the nodal points x_i. Also, $\mathcal{E}_k$ can be viewed as a linear operator $S_h^0(\Omega) \to S_h^0(\Omega)$ with a trivial modification of the above definition, namely

$$\mathcal{E}_k\phi(x_i) = \begin{cases} \phi(x_i) & \text{for } x_i \in \partial\Omega_k, \\ 0 & \text{for } x_i \in \Omega \setminus \partial\Omega_k. \end{cases}$$

We shall use $\mathcal{E}_k$ in both contexts since it will be easy to determine which is the right one from the functions $\mathcal{E}_k$ is applied to.

Similarly, let $\mathcal{E} : S_h^0(\Omega) \mapsto S_h^0(\Omega)$ be defined by

$$\mathcal{E}\phi(x_i) = \begin{cases} \phi(x_i) & \text{for } x_i \in \Gamma, \\ 0 & \text{for } x_i \in \Omega \setminus \Gamma. \end{cases} \tag{10}$$

For each k, let $\mathcal{B}_k(\cdot,\cdot)$ be a bilinear form on $S_h^0(\Omega_k) \times S_h^0(\Omega_k)$ which is uniformly equivalent to $\mathcal{A}_k(\cdot,\cdot)$, where $\mathcal{A}_k(\cdot,\cdot)$ is defined as in (3) but with integration only on Ω_k. By this we mean that for each k there are constants c_k and C_k with C_k/c_k bounded independently of h and d such that

$$c_k\mathcal{B}_k(V,V) \leq \mathcal{A}_k(V,V) \leq C_k\mathcal{B}_k(V,V), \quad \text{for all} \quad V \in S_h^0(\Omega_k). \tag{11}$$

The preconditioning form is given by

$$\begin{aligned}\mathcal{B}(U,V)=\sum_{k=1}^{n_d}&\mathcal{B}_k(U-\bar{U}_k-\mathcal{E}_k(U-\bar{U}_k),V-\bar{V}_k-\mathcal{E}_k(V-\bar{V}_k))\\&+h^{-1}\sum_{k=1}^{n_d}\tilde{a}_k\langle U-\bar{U}_k,V-\bar{V}_k\rangle_{\partial\Omega_k,h}.\end{aligned}\tag{12}$$

Here, $\bar{U}_k$ denotes the discrete mean value of U on $\partial\Omega_k$, i.e.,

$$\bar{U}_k\equiv\frac{\langle U,1\rangle_{\partial\Omega_k,h}}{\langle 1,1\rangle_{\partial\Omega_k,h}}.$$

In (12), $\tilde{a}_k,\ k=1,\dots,n_d$ are parameters. For example, if $\tilde{a}_k$ is taken to be the smallest eigenvalue of $\{a_{i,j}\}$ at some point $x\in\Omega_k$ then

$$C_k^{-1}\tilde{a}_k\mathcal{D}_k(v,v)\le\mathcal{A}_k(v,v)\le C_k\tilde{a}_k\mathcal{D}_k(v,v),\quad\text{for all}\quad v\in S_h(\Omega_k),\tag{13}$$

where C_k depends only on the local variation of the coefficients $\{a_{ij}\}$ on the subdomain Ω_k. Consequently, we will assume that (13) holds with C_k/c_k bounded independently of d, h, and k.

We introduce some standard assumptions about the domain Ω, the subdomain splitting and the associated finite element spaces which are needed for the analysis.

We start by requiring that the collection $\{\Omega_k\}$ be quasi-uniform of size d. Also, we shall assume that

$$|u|^2_{\partial\Omega_k}\le C\{\epsilon^{-1}\|u\|^2_{\Omega_k}+\epsilon\mathcal{D}_k(u,u)\},\tag{14}$$

holds for any ϵ in $(0,d]$ and all k. Finally, we assume that a Poincaré inequality of the form

$$\|v\|^2_{\Omega_k}\le Cd^2\mathcal{D}_k(v,v)\tag{15}$$

holds for functions v with zero mean value on Ω_k.

The inequalities (14) and (15) hold for all but pathological subdomains. A sufficient but by no means necessary condition for the above two inequalities is given in the following assumption.

Each Ω_k is star-shaped with respect to a point. This means that for each Ω_k there is a point $\hat{x}_k$ and a constant $c_k>0$ such that $(x-\hat{x}_k)\cdot\mathbf{n}(x)\ge c_kd$ for all $x\in\partial\Omega_k$ which are not mesh vertices. We further assume that $c_k\ge c$ for some constant c not depending on d, k or h. Here $\mathbf{n}(x)$ denotes the outward unit normal vector to $\partial\Omega_k$ at a nonvertex point x.

The following theorem establishes bounds for the asymptotic behavior of the preconditioner $\mathcal{B}(\cdot,\cdot)$.

Theorem 3.1 *Let $\mathcal{A}(\cdot,\cdot)$ and $\mathcal{B}(\cdot,\cdot)$ be given by* (3) *and* (12)*, respectively. Then there exist positive constants c and C not depending on d or h such that*

$$c\mathcal{A}(V,V)\le\mathcal{B}(V,V)\le C\frac{d}{h}\mathcal{A}(V,V),\tag{16}$$

for all $V\in S_h^0(\Omega)$.

Remark 3.1 *The preconditioning form $\mathcal{B}(\cdot,\cdot)$ defined above is not uniformly equivalent to $\mathcal{A}(\cdot,\cdot)$. Nevertheless, its preconditioning effect is very close to that of a uniform preconditioner for many practical problems, particularly in three spatial dimensions. The number of subdomains often equals the number of processors in a parallel implementation and it is now feasible to keep d on the order of $h^{1/2}$. Applying a conjugate gradient method preconditioned by $\mathcal{B}(\cdot,\cdot)$ for solving* (5) *would result in a number of iterations proportional to $h^{-1/4}$. In $\mathbb{R}^3$, if Ω is the unit cube, $h = 10^{-2}$ corresponds to a very large computational problem whereas $10^{1/2} \approx 3.2$. Also, it is well known that classical overlapping domain decomposition algorithms with small overlap exhibit the same condition number growth but in contrast to our method the overlapping preconditioners are adversely sensitive to large jumps in the operator coefficients (see Remark 3.3 below).*

Remark 3.2 *The constants c and C in Theorem 3.1 depend on the local (with respect to the subdomains) behavior of the operator and the preconditioner. Clearly, one of the most influential factors on the local properties of $\mathcal{A}(\cdot,\cdot)$ and $\mathcal{B}(\cdot,\cdot)$ is the coefficient matrix $\{a_{i,j}\}|_{\Omega_k}$. In fact, the constants C_k in* (13) *depend on the local lower and upper bounds for the eigenvalues of $\{a_{i,j}\}|_{\Omega_k}$ and in general so do the constants c_k and C_k in* (11). *Therefore, in applications to problems with large jumps in the coefficients, it is desirable, if possible, to align the subdomain boundaries with the locations of the jumps. In this case the preconditioner* (12) *will be independent of these jumps.*

Remark 3.3 *The utilization of the averages $\bar{U}_k$ plays the role of a coarse problem especially designed to take into account cases with interior subdomains and also applications with large jumps in the operator coefficients, provided that the locations of the jumps are aligned with the subdomain boundaries. To illustrate that the role of the averages in overcoming difficulties coming from large jumps of the coefficients is essential, we consider a conventional additive Schwarz preconditioner with minimal overlap [DW94]. The asymptotic condition number bound provided in [DW94] is the same as that of our theorem in the case of smooth coefficients. However, because of the deterioration in the approximation and boundedness properties of the weighted L^2 projection into the coarse subspace [BX91], the condition number of the preconditioned system for the minimal overlap algorithm when $n = 3$ can only be bounded by $(d/h)^2$.*

Our preconditioner is very economical computationally. In fact, it allows the use of efficient subdomain preconditioners such as one multigrid V-cycle (cf. [Bra93]). The use of the simple extension $\mathcal{E}$ also results in enhanced efficiency.

4 An Additive Schwarz Reformulation of the Domain Decomposition Algorithm

A very important observation for the subsequent analysis is that the preconditioner $\mathcal{B}(\cdot,\cdot)$ can be viewed as an additive subspace correction method (cf. [BPX90] and [Xu92]) with judiciously chosen subspaces. Let the linear operator $\tilde{\mathcal{E}} : S_h^0(\Omega) \mapsto S_h^0(\Omega)$ be defined by

$$\tilde{\mathcal{E}}V = \mathcal{E}V + \sum_{k=1}^{n_d}(\bar{V}_k - \mathcal{E}_k\bar{V}_k).$$

In the above definition, $\bar{V}_k$ is a constant function with support in the closed subdomain Ω_k.

Furthermore, define

$$\hat{S}_h^0(\Omega) = \{v \in S_h^0(\Omega) \mid v = 0 \quad \text{on}\, \Gamma\}$$

and

$$S_\Gamma(\Omega) = \{\tilde{\mathcal{E}}v \mid v \in S_h^0(\Omega)\}.$$

Thus $\hat{S}_h^0(\Omega)$ and $S_\Gamma(\Omega)$ provide a direct sum decomposition of $S_h^0(\Omega)$.

The additive Schwarz preconditioner applied to $g \in S_h^0(\Omega)$ based on the above two spaces results in a function $Y = Y_0 + Y_\Gamma$ where $Y_0 \in \hat{S}_h^0(\Omega)$ satisfies

$$\mathcal{B}_0(Y_0, \phi) = (g, \phi), \text{ for all } \phi \in \hat{S}_h^0(\Omega) \tag{17}$$

and $Y_\Gamma \in S_\Gamma(\Omega)$ satisfies

$$\mathcal{B}_\Gamma(Y_\Gamma, \phi) = (g, \phi), \text{ for all } \phi \in S_\Gamma(\Omega). \tag{18}$$

Here $\mathcal{B}_0(\cdot,\cdot)$ and $\mathcal{B}_\Gamma(\cdot,\cdot)$ are symmetric and positive definite bilinear forms.

We shall see that the preconditioner in (12) is equivalent to the additive Schwarz method above when

$$\mathcal{B}_0(\varphi, \phi) = \sum_{k=1}^{n_d} \mathcal{B}_k(\varphi, \phi) \tag{19}$$

and

$$\mathcal{B}_\Gamma(\varphi, \phi) = h^{-1} \sum_{k=1}^{n_d} \tilde{a}_k \langle \varphi - \bar{\varphi}_k, \phi - \bar{\phi}_k \rangle_{\partial\Omega_k, h}. \tag{20}$$

Let W be the solution of (8). Then

$$\mathcal{B}(W, \varphi) = \mathcal{B}_k(W^{(k)}, \varphi) = (g, \varphi)_\Omega, \quad \text{for all} \quad \varphi \in S_h^0(\Omega_k), \tag{21}$$

where $W^{(k)} \equiv W - \bar{W}_k - \mathcal{E}_k(W - \bar{W}_k)$. The function Y_0 satisfying (17) is given by

$$Y_0 = W - \tilde{\mathcal{E}}W \quad \text{on} \quad \Omega_k.$$

The form given by (20) depends only on the boundary values of φ and ϕ. Also, the function Y_Γ solving (18) equals the solution W on Γ. From the definition of $\tilde{\mathcal{E}}$,

$$Y_\Gamma = \tilde{\mathcal{E}}W = \mathcal{E}W + \sum_{k=1}^{n_d} (\bar{W}_k - \mathcal{E}_k \bar{W}_k).$$

Thus, the solution W of (8) is the result of the additive Schwarz algorithm with subspace decomposition given by $\hat{S}_h^0(\Omega)$ and $S_\Gamma(\Omega)$, with forms defined by (19) and (20).

5 Alternative Inexact Additive Preconditioners

We now consider a classical technique for developing nonoverlapping domain decomposition preconditioners. The behavior of such methods has been investigated in the case when the boundary form is uniformly equivalent to the corresponding Schur complement subsystem [B̈89], [HLM91]. Here, we show that this method also reduces to an additive Schwarz preconditioner. In addition, we show that the inexact solve technique combined with the boundary form discussed earlier provides an effective preconditioner. Indeed, our results are much better than what would be expected from the analysis of [B̈89], [HLM91].

The classical inexact domain decomposition preconditioners are easily understood from the matrix point of view. In this case, one orders the unknowns so that the stiffness matrix corresponding to $\mathcal{A}(\cdot,\cdot)$ can be written in a block form as

$$\begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix}.$$

Here $\mathbf{A}_{22}$ corresponds to the nodes on Γ and $\mathbf{A}_{11}$ to the remaining nodes. With this ordering, the form corresponding to a typical domain decomposition preconditioner (e.g., [BPS86a],[BPS87],[BPS88], [BPS89]) has a stiffness matrix of the form

$$\hat{\mathbf{A}} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{Z} \end{pmatrix},$$

where $\mathbf{Z} = \mathbf{B}_{22} + \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}$ and $\mathbf{B}_{22}$ is the domain decomposition boundary preconditioning matrix. Inverting $\hat{\mathbf{A}}$ is a three step block Gaussian elimination procedure.

The classical inexact method is defined by replacing $\mathbf{A}_{11}$ with $\mathbf{B}_{11}$ where $\mathbf{B}_{11}$ is another symmetric and positive definite matrix. This defines a new preconditioning operator $\mathbf{B}$ given by

$$\mathbf{B} = \begin{pmatrix} \mathbf{B}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \tilde{\mathbf{Z}} \end{pmatrix}. \tag{22}$$

Here $\tilde{\mathbf{Z}}$ is given by $\tilde{\mathbf{Z}} = \mathbf{B}_{22} + \mathbf{A}_{21}\mathbf{B}_{11}^{-1}\mathbf{A}_{12}$.

Generally, the inexact algorithm may not converge as well as the exact version. Even if one takes $\mathbf{B}_{22}$ to be the Schur complement, $\mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{B}_{11}^{-1}\mathbf{A}_{12}$, the inexact preconditioner may perform poorly unless the difference between the two matrices $\mathbf{B}_{11}$ and $\mathbf{A}_{11}$ is sufficiently small in an appropriate sense (see Theorem 5.1).

We now show that the inexact preconditioners correspond to additive Schwarz methods. The first subspace in this decomposition is $\hat{S}_h^0(\Omega)$. Let $\mathcal{B}_0(\cdot,\cdot)$ be the form on $\hat{S}_h^0(\Omega) \times \hat{S}_h^0(\Omega)$ with stiffness matrix $\mathbf{B}_{11}$. The second subspace is given by

$$\begin{aligned} \hat{S}_h(\Gamma) = \Big\{ & \mathcal{E}\varphi + \varphi_0 \,|\, \varphi \in S_h^0(\Omega); \\ & \mathcal{B}_0(\varphi_0, \phi) = -\mathcal{A}(\mathcal{E}\varphi, \phi), \text{ for all } \phi \in \hat{S}_h^0(\Omega) \Big\}. \end{aligned} \tag{23}$$

Clearly, the functions in $\hat{S}_h(\Gamma)$ are completely determined by their traces on Γ. Let $\mathcal{B}_\Gamma(\cdot,\cdot)$ be the form on $\hat{S}_h(\Gamma)\times\hat{S}_h(\Gamma)$ with stiffness matrix $\mathbf{B}_{22}$. $\mathcal{B}_\Gamma(u,v)$ depends only on the boundary nodal values of u and v and thus naturally extends to $S_h^0(\Omega)\times S_h^0(\Omega)$.

Clearly, $\hat{S}_h^0(\Omega)$ and $\hat{S}_h(\Gamma)$ provide a direct sum decomposition of $S_h^0(\Omega)$. This decomposition is tied strongly to the bilinear form $\mathcal{B}_0(\cdot,\cdot)$. In particular, if $\mathcal{B}_0(\cdot,\cdot)\equiv\mathcal{A}(\cdot,\cdot)$ on $\hat{S}_h^0(\Omega)\times\hat{S}_h^0(\Omega)$ then the space $\hat{S}_h(\Gamma)$ consists of discrete harmonic functions and the decomposition is $\mathcal{A}(\cdot,\cdot)$–orthogonal. In general, the decomposition is not $\mathcal{A}(\cdot,\cdot)$–orthogonal.

The preconditioner defined by (22) can be restated as an operator $\mathbf{B}: S_h^0(\Omega)\mapsto S_h^0(\Omega)$. In fact, it is a straightforward exercise to check that it corresponds to the preconditioning operator defined in the following algorithm.

Algorithm 5.1 *Given $g\in S_h^0(\Omega)$ we define $\mathbf{B}^{-1}g=U$ where U is computed as follows:*

1. *Compute $U_0\in\hat{S}_h^0(\Omega)$ by solving*

$$\mathcal{B}_0(U_0,\varphi)=(g,\varphi)\quad \textit{for all}\quad \varphi\in\hat{S}_h^0(\Omega). \tag{24}$$

2. *Compute the trace U_Γ on Γ by solving*

$$\mathcal{B}_\Gamma(U_\Gamma,\mathcal{E}\phi)=(g,\mathcal{E}\phi)-\mathcal{A}(U_0,\mathcal{E}\phi)\quad \textit{for all}\quad \phi\in\hat{S}_h(\Gamma).$$

3. *Compute $U_{\Gamma 0}$ by solving*

$$\mathcal{B}_0(U_{\Gamma 0},\varphi)=-\mathcal{A}(\mathcal{E}U_\Gamma,\varphi)\quad \textit{for all}\quad \varphi\in\hat{S}_h^0(\Omega).$$

4. *Set $U=U_0+\mathcal{E}U_\Gamma+U_{\Gamma 0}$.*

Although the above algorithm appears as a multiplicative procedure, we shall now demonstrate that it is equivalent to an additive Schwarz method. It is easy to see that the problem solved in Step 2 of Algorithm 5.1 is independent of U_0. Indeed, for any $\phi\in\hat{S}_h(\Gamma)$, we decompose $\phi=\mathcal{E}\phi+\phi_0$ as in (23) and observe

$$-\mathcal{A}(\mathcal{E}\phi,U_0)=\mathcal{B}_0(\phi_0,U_0)=(g,\phi_0).$$

Thus, Steps 2 and 3 of the above algorithm reduce to finding $U_\Gamma\in\hat{S}_h(\Gamma)$ such that

$$\mathcal{B}_\Gamma(U_\Gamma,\phi)=(g,\phi)\quad \text{for all}\quad \phi\in\hat{S}_h(\Gamma). \tag{25}$$

Hence, $\mathbf{B}^{-1}g=U=U_0+U_\Gamma$ where U_0 and U_Γ satisfy (24) and (25) respectively, i.e., Algorithm 5.1 is an implementation of an additive Schwarz procedure.

Notice that Algorithm 5.1 avoids the need of knowing explicitly a basis for the space $\hat{S}_h(\Gamma)$ which could be either a computationally expensive problem or a significant complication of the overall algorithm. Obviously this procedure provides inexact variants of the methods given in [BPS86a], [BPS87], [BPS88], and [BPS89].

It follows that the preconditioning form $\mathcal{B}(\cdot,\cdot)$ corresponding to the operator defined in Algorithm 5.1 is given by

$$\mathcal{B}(V,V)=\mathcal{B}_0(V_0,V_0)+\mathcal{B}_\Gamma(V_\Gamma,V_\Gamma). \tag{26}$$

Here $V = V_0 + V_\Gamma$ with $V_0 \in \hat{S}_h^0(\Omega)$ and $V_\Gamma \in \hat{S}_h(\Gamma)$.

In the remainder of this section we provide bounds for (26). We take

$$\mathcal{B}_0(u,v) = \sum_{k=1}^{n_d} \mathcal{B}_k(u,v)$$

where $\mathcal{B}_k(\cdot,\cdot)$ is defined as in Section 3 (with C_k/c_k in (11) bounded independently of h, k, and d). The first theorem in this section was given by Börgers [B8̈9] and Haase at al. [HLM91] and provides a result when $\mathbf{B}_{22}$ is uniformly equivalent to the Schur complement $\mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}$. This is the same as assuming that the quadratic form $\mathcal{B}_\Gamma(\cdot,\cdot)$ is equivalent to the boundary form with diagonal

$$\inf_{\phi \in \hat{S}_h^0(\Omega)} A(u+\phi, u+\phi), \quad \text{for all} \quad u \in \hat{S}_h(\Gamma). \tag{27}$$

Theorem 5.1 *Let $\mathcal{A}(\cdot,\cdot)$ be given by* (3) *and $\mathcal{B}(\cdot,\cdot)$ by* (26)*. Assume that the quadratic form $\mathcal{B}_\Gamma(\cdot,\cdot)$ is uniformly equivalent to the quadratic from induced by* (27)*. In addition, let γ be the smallest positive constant such that*

$$|\mathcal{A}(\varphi,\varphi) - \mathcal{B}(\varphi,\varphi)| \le \gamma \mathcal{A}(\varphi,\varphi) \quad \textit{for all} \quad \varphi \in \hat{S}_h^0(\Omega). \tag{28}$$

Then

$$c\left(\frac{\gamma^2}{h}\right)^{-1} \mathcal{A}(U,U) \le \mathcal{B}(U,U) \le C\frac{\gamma^2}{h}\mathcal{A}(U,U)$$

holds for all $U \in S_h^0(\Omega)$ with constants c and C independent of d and h.

Remark 5.1 *Condition* (28) *requires that $\mathcal{B}_0(\cdot,\cdot)$ should be a good approximation to $\mathcal{A}(\cdot,\cdot)$ for the preconditioner* (26) *to be efficient. The result of the theorem shows that if* (28) *holds with γ on the order of $h^{1/2}$ then the preconditioner $\mathcal{B}(\cdot,\cdot)$ is uniform. However, the development of a form $\mathcal{B}_0(\cdot,\cdot)$ satisfying* (28) *usually involves significant additional computational work since γ must tend to zero as h becomes small. Alternatively keeping γ fixed independent of h may result in a rather ill-conditioned method when h is small. However, there are examples of reasonably accurate preconditioners $\mathcal{B}_0(\cdot,\cdot)$, e.g. multigrid V- or W-cycles, which appear to perform well when h is not very small (cf. [B8̈9]) due to the fact that the corresponding γ's are comparable to $h^{1/2}$.*

The main result of this section is given in the next theorem. It is for the case when

$$\mathcal{B}_\Gamma(u,v) = h^{-1}\sum_{k=1}^{n_d} \tilde{a}_k\langle u - \bar{u}_k, v - \bar{v}_k\rangle_{\partial\Omega_k,h}, \quad \text{for all} \quad u,\, v \in \hat{S}_h(\Gamma). \tag{29}$$

Theorem 5.2 *Let $\mathcal{A}(\cdot,\cdot)$ be given by* (3)*, $\mathcal{B}(\cdot,\cdot)$ be given by* (26)*, and $\mathcal{B}_\Gamma(\cdot,\cdot)$ defined by* (29)*. Then*

$$c\mathcal{A}(U,U) \le \mathcal{B}(U,U) \le C\frac{d}{h}\mathcal{A}(U,U) \tag{30}$$

holds for all $U \in S_h^0(\Omega)$ with constants c and C independent of d and h.

Remark 5.2 *The result of Theorem 5.2 shows that introducing inexact solves in the interior of the subdomains does not deteriorate the overall preconditioning effect of the corresponding exact method analyzed in [BPS87]. As we have pointed out in Remark 3.1, the adverse effect of h approaching zero on the condition number can be compensated for easily by adjusting the parameter d. This balance is an alternative to* (28) *and could be a better choice when h is small relative to γ. In fact, the utilization of the bilinear form* (29) *leads to computationally efficient algorithms, unconstrained by accuracy conditions like* (28). *The differences in the preconditioning effect of the inexact (Algorithm 5.1) and exact (cf. [BPS87]) methods are negligible. However, the savings of computational time are significant in favor of Algorithm 5.1.*

REFERENCES

[B̈89] Börgers C. (1989) The Neumann–Dirichlet domain decomposition method with inexact solvers on the subdomains. *Numer. Math.* 55: 132–136.

[BPS86a] Bramble J., Pasciak J., and Schatz A. (1986) The construction of preconditioners for elliptic problems by substructuring, I. *Math. Comp.* 47: 103–134.

[BPS86b] Bramble J., Pasciak J., and Schatz A. (1986) An iterative method for elliptic problems on regions partitioned into substructures. *Math. Comp.* 46: 361–369.

[BPS87] Bramble J., Pasciak J., and Schatz A. (1987) The construction of preconditioners for elliptic problems by substructuring, II. *Math. Comp.* 49: 1–16.

[BPS88] Bramble J., Pasciak J., and Schatz A. (1988) The construction of preconditioners for elliptic problems by substructuring, III. *Math. Comp.* 51: 415–430.

[BPS89] Bramble J., Pasciak J., and Schatz A. (1989) The construction of preconditioners for elliptic problems by substructuring, IV. *Math. Comp.* 53: 1–24.

[BPWX91] Bramble J., Pasciak J., Wang J., and Xu J. (1991) Convergence estimates for product iterative methods with applications to domain decomposition. *Math. Comp.* 57: 1–21.

[BPX90] Bramble J., Pasciak J., and Xu J. (1990) Parallel multilevel preconditioners. *Math. Comp.* 55: 1–22.

[BPX91] Bramble J., Pasciak J., and Xu J. (1991) A multilevel preconditioner for domain decomposition boundary systems. In *Proceedings of the 10'th International Conference on Computational Methods in Applied Sciences and Engineering.* Nova Sciences, New York.

[Bra93] Bramble J. (1993) *Multigrid Methods*, volume 294 of *Pitman Research Notes in Mathematics Series.* Longman Scientific & Technical, London.

[BW86] Bjørstad P. E. and Widlund O. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal.* 23: 1097–1120.

[BX91] Bramble J. and Xu J. (1991) Some estimates for weighted L^2 projections. *Math. Comp.* 56: 463–476.

[CMW95] Cowsar L., Mandel J., and Wheeler M. (1995) Balancing domain decomposition for mixed finite elements. *Math. Comp.* 64: 989–1015.

[Dry82] Dryja M. (1982) A capacitance matrix method for the Dirichlet problem on a plygonal region. *Numer. Math.* 39: 51–64.

[Dry88] Dryja M. (1988) A method of domain decomposition for three-dimensional finite element elliptic problems. In Glowinski R., Golub G., Meurant G., and Périaux J. (eds) *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 43–61. SIAM, Philadelphia, PA.

[DSW94] Dryja M., Smith B., and Widlund O. (1994) Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions. *SIAM J. Numer. Anal.* 31(6): 1662–1694.

[DW91] Dryja M. and Widlund O. (1991) Additive Schwarz methods for elliptic finite element problems in three dimensions. Technical Report 570, Courant Institute of Mathematical Sciences, New York, NY.

[DW94] Dryja M. and Widlund O. (1994) Domain decomposition algorithms with small overlap. *SIAM J. Sci. Comp.* 15: 604–620.

[GW87] Gonzalez R. and Wheeler M. (1987) Domain decomposition for elliptic partial differential equations with neumann boundary conditions. *Parallel Comput.* 5: 257–263.

[HLM91] Haase G., Langer U., and Meyer A. (1991) The approximate Dirichlet domain decomposition method. Part I: An algebraic approach. *Computing* 47: 137–151.

[Nep91] Nepomnyaschikh S. (1991) Application of domain decomposition to elliptic problems with discontinuous coefficients. In Glowinski R., Kuznetzov Y., Meurant G., and Périaux J. (eds) *Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 242–251. SIAM, Philadelphia, PA.

[BDV96] Bjørstad P. E., Dryja M., and Vainikko E. (1996) Additive schwarz methods without subdomain overlap and with new coarse spaces. In Glowinski R., Périaux J., Shi Z., and Widlund O. (eds) *Domain Decomposition Methods in Science and Engineering.*

[Smi90] Smith B. (1990) *Domain Decomposition Algorithms for the Partial Differential Equations of Linear Elasticity.* PhD thesis, Courant Institute of Mathematical Sciences, New York, NY.

[Xu92] Xu J. (1992) Iterative methods by space decomposition and subspace correction. *SIAM Review* 34: 581–613.

6

A New Approach to Domain Decomposition Methods with Non-matching Grids

Abdellatif Agouzal and Naïma Debit

1 Introduction

Attempts at solving actual problems, e.g. heterogeneous equations, have revealed limitations of many classical domain decomposition methods. As a result, there has been renewed interest in new and alternative approaches.

In the context of non-matching grids, to our knowledge, three approaches are considered in literature: Mortar element methods, in *primal formulation* ([AMMP90],[BDM90],[BMP92]), or *mixed or equilibrium formulation* ([Ago96]); hybrid methods ([AT95],[RT97]); and primal-equilibrium coupling methods ([AL94a]). Mortar element type methods are based on the explicit construction of an approximation space. The approach we present here is a conforming one, in which no global approximation is constructed. The domain is decomposed in two block-subdomains allowing for internal subdomain decomposition. A primal variational formulation is used in one region, whereas an equilibrium one is used on the other. The flexibility of the method allows for use of different discretizations on each subdomain; of low-order type (e.g. finite element methods [Cia91]) or high-order type (e.g. spectral element methods [CHQZ88]). We will use in this paper either finite element or spectral element versions of the method. The solution is discontinuous on the interface, and the matching is implicitly contained in the equations formulation.

The main characteristics of the approach we introduce can be summarized as:

- Flexibility on the choice of discretizations on each subdomain;
- No global discrete space to contruct : The global space is a product of local ones; the solution is *"discontinuous"* on the interface;
- No Lagrange multiplier is used to take into account the constraint on the interface.

This paper describes recent advances in the development of the present approach. We give here the main results and leave the detailed analysis for related papers.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 Problem Formulation

The Continuous Case

The discussion here is restricted to second-order linear partial differential equations. We consider the solution of the Poisson equation on a domain Ω : Find u such that

$$\begin{cases} Lu &= -\Delta u + u = f \quad \text{in } \Omega, \\ u &= 0 \quad\quad\quad\quad\quad\ \ \text{on } \Gamma = \partial\Omega. \end{cases}$$

where $f \in L^2(\Omega)$.

Remark *In all what follows, L can be replaced by $Lu = -div(K$ **grad** $u) + \beta.$ **grad** $u + \sigma u$, provided that all corresponding problem is satisfy standard solvability hypotheses.*

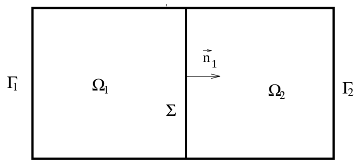
We suppose (for simplicity) that Ω is rectangularly decomposable, that is, there exist rectangular subdomains Ω_1 and Ω_2 such that

$$\overline{\Omega} = \overline{\Omega_1} \cup \overline{\Omega_2}\ ,\ \ \Omega_1 \cap \Omega_2 = \emptyset.$$

In the sequel, we set

$$\Sigma = \partial\Omega_1 \cap \partial\Omega_2.$$

Figure 1



First, we remark that this Problem can be factored to give the coupled system :

$$\begin{aligned} \textbf{grad}\ u_1 &= p_1 \ \text{ in } \Omega_1, \\ -div p_1 + u_1 &= f \ \text{ in } \Omega_1, \\ -\Delta u_2 + u_2 &= f \ \text{in } \Omega_2, \\ u_1 &= u_2 \ \text{ on } \Sigma, \\ u_1 = 0 \ \text{ on } \Gamma_1 &= \partial\Omega_1/\Sigma, \\ u_2 = 0 \ \text{ on } \Gamma_2 &= \partial\Omega_2/\Sigma, \\ p_1.n_1 + \frac{\partial u_2}{\partial n_2} &= 0 \ \text{ on } \Sigma \end{aligned} \tag{1}$$

where $\frac{\partial}{\partial n_2}$ is the outward normal derivative, and $p_1.n_1$ is the outward normal trace of p.

In the framework of the numerical solution of (1) by finite element type or spectral element type methods [Cia91], it is essential to work in a suitable variational context. Otherwise stated, one has to use variational forms leading to a well-posed problem equivalent to system (1) in a given sense.

The weak form of (1) is given by seeking a pai $(p_1, u_2) \in H(div, \Omega_1) \times H^1_{\Gamma_2,0}(\Omega_2)$ such that :

$$\begin{array}{ll} \forall q_1 \in H(div, \Omega_1), & \displaystyle\int_{\Omega_1} (p_1.q_1 + divp_1\ divq_1)dx - < q_1.n_1, u_2 >_\Sigma = -\int_{\Omega_1} f\ divq_1 dx, \\ \forall v_2 \in H^1_{\Gamma_2,0}(\Omega_2), & \displaystyle\int_{\Omega_2} \{\ \mathbf{grad}\ u_2.\ \mathbf{grad}\ v_2 + u_2 v_2\}dx + < p_1.n_1, v_2 >_\Sigma = \int_{\Omega_2} f v_2 dx \end{array} \tag{2}$$

where $< .,. >_\Sigma$ is the duality pairing between the function spaces $H^{\frac{1}{2}}_{0,0}(\Sigma)$ and $H^{-\frac{1}{2}}(\Sigma)$, and

$$H^1_{0,\Gamma_2}(\Omega_2) = \{v \in H^1(\Omega_2);\ \text{such that } v = 0 \text{ on } \Gamma_2\}.$$

Note that the unknowns p_1 and u_2 are coupled only through the boundary integrals appearing in (2).

One can also write the problem (2) in another useful form, namely
Find $(p_1, u_2) \in H(div, \Omega_1) \times H^1_{\Gamma_2,0}(\Omega_2)$ such that,
$\forall (q_1, v_2) \in H(div, \Omega_1) \times H^1_{\Gamma_2,0}(\Omega_2)$,

$$B((p_1, u_2); (q_1, v_2)) = -\int_{\Omega_1} f\ divq_1 dx + \int_{\Omega_2} f v_2 dx \tag{3}$$

where

$$\begin{array}{rcl} B((p_1, u_2); (q_1, v_2)) & = & \int_{\Omega_1} (p_1.q_1 + divp_1\ divq_1)dx - < q_1.n_1, u_2 >_\Sigma \\ & + & \int_{\Omega_2} \{\ \mathbf{grad}\ u_2.\ \mathbf{grad}\ v_2 + u_2 v_2\}dx + < p.n_1, v_2 >_\Sigma \end{array}$$

Concerning the existence and uniqueness results, we have

Theorem 2.1 *There exists a unique solution (p_1, u_2) of problem (2). Moreover,*

$$\begin{array}{rl} p_1 = & \mathbf{grad}\ u_{|\Omega_1}, \\ & u_2 = u_{|\Omega_2}, \end{array} \tag{4}$$

where u is the weak solution of the Helmholtz problem (2.1).

Proof: First remark that the bilinear form B(.;.) is $H(div, \Omega_1) \times H^1_{0,\Gamma_2}(\Omega_2)$-elliptic. We prove easily the continuity of this form. So by Lax-Milgram theorem, problem (2) has a unique solution. The second part of the theorem is obtained by a slight modification of standard arguments.

The Approximated Problem

For its numerical solution, the variational problem (2) must first be approximated by a problem with a finite number of unknowns [Cia91]. In the finite element or spectral methods context, this approximation is realized by replacing the space $H(div, \Omega_1) \times H^1_{0,\Gamma_2}(\Omega_2)$ by a finite dimensional space. In this method, we want to approximate separately the spaces $H(div, \Omega_1)$ and $H^1_{0,\Gamma_2}(\Omega_2)$. Therefore, we introduce finite dimensional spaces :

$$V_{h_1} \subset H(div, \Omega_1), \qquad \dim\, V_{h_1} < +\infty$$
$$V_{h_2} \subset H^1_{\Gamma_2,0}(\Omega_2), \qquad \dim V_{h_2} < +\infty.$$

The classical conforming Galerkin approximation of (2) is
Find $(p_{h_1}, u_{h_2} \in V_{h_1} \times V_{h_2}$ such that ,
$\forall (q_{h_1}, v_{h_2}) \in V_{h_1} \times V_{h_2}$,

$$B((p_{h_1}, u_{h_2}); (q_{h_1}, v_{h_2})) = -\int_{\Omega_1} f \; divq_{h_1} dx + \int_{\Omega_2} f v_{h_2} dx. \qquad (5)$$

Similarly to problem (2), problem (5) has a unique solution. Moreover, it is possible to prove the following

Theorem 2.2 *Let $(p_1, u_2) \in H(div, \Omega_1) \times H^1_{0,\Gamma_2}(\Omega_2)$ be the solution of (2) and let V_{h_1} and V_{h_2} defined as above. Problem (5) has a unique solution (p_{h_1}, u_{h_2}) and there exists a constant C which does not depend on dimensions of V_{h_1} and V_{h_2} such that*

$$\begin{aligned} \|p_1 - p_{h_1}\|_{H(div,\Omega_1)} &+ \quad \|u_2 - u_{h_2}\|_{1,\Omega_2} \leq \\ C &\quad \inf_{(q_{h_1}, v_{h_2}) \in V_{h_1} \times V_{h_2}} \{ \|p_1 - q_{h_1}\|_{H(div,\Omega_1)} + \|u_2 - v_{h_2}\|_{1,\Omega_2} \}. \end{aligned}$$

Proof: Follows easily from Lax-Milgram Theorem and Céa Lemma.

An Example of Discretization

A basic choice of V_{h_1} and V_{h_2} in the spectral methods context consists of introducing:

$$V_{h_1} = RT_{N1}(\Omega_1) = P_{N1,N1-1}(\Omega_1) \times P_{N1-1,N1}(\Omega_1)$$

and

$$V_{h_2} = Q_{N_2}(\Omega_2) \cap H^1_{0,\Gamma_2}(\Omega_2).$$

With this choice, a consequence of (2.2) is the following

Theorem 2.3 *Assume that the solution (p_1, u_2) of problem (2) is such that, p_1 and $divp_1$ belong to $(H^{\sigma_1}(\Omega_1))^d$ and $H^{\sigma_1}(\Omega_1)$ for a real number $\sigma_1 > 0$, and u_2 belongs to $H^{\sigma_2}(\Omega_2)$ for a real number σ_2, $1 < \sigma_2$. Then the following estimate holds*

$$\begin{aligned} \|p_1 - p_{h_1}\|_{H(div,\Omega_1)} &+ \quad \|u_2 - u_{h_2}\|_{1,\Omega_2} \leq \\ C_\epsilon &\quad \{ N_1^{-\sigma_1+\epsilon} (\|p_1\|_{\sigma_1,\Omega_1} + \|divp_1\|_{\sigma_1,\Omega_1}) + N_2^{-\sigma_2+1} \|u_2\|_{\sigma_2,\Omega_2} \}. \end{aligned}$$

for all $\epsilon > 0$.

By post-processing, we can easily obtain an approximation of u_1. More precisely, if we set

$$u_{N_1} = \Pi_{N_1-1}(divp_{N_1} + f)$$

where Π_{N_1-1} is the projection operator defined from $L^2(\Omega_1)$ onto $P_{N_1-1}(\Omega_1)$, we have

$$\|u_1 - u_{N_1}\|_{0,\Omega_1} \leq C_\epsilon \{ N_1^{-\sigma_1+\epsilon} (\|p_1\|_{\sigma_1,\Omega_1} + \|divp_1\|_{\sigma_1,\Omega_1}) + N_2^{-\sigma_2+1} \|u_2\|_{\sigma_2,\Omega_2} \}.$$

for all $\epsilon > 0$.
These results are illustrated in the following figure.

Figure 2 We consider here the solution of problem (2.1) on the domain $\Omega = (-1, 1)^2$. A spectral element discretization is used on each subdomain. The right-hand side f is given by the exact solution $u_{ex}(x, y) = \sin(\pi x) \sin(\pi x)$. We plot in **(a)** the H^1 - error of u in Ω_2 as a function of related polynomial order; and in **(b)** the $L^2 - error$ of u in Ω_1, obtained by post-processing. The error decreases exponentially fast as would be expected for spectral approximation of a smooth solution.

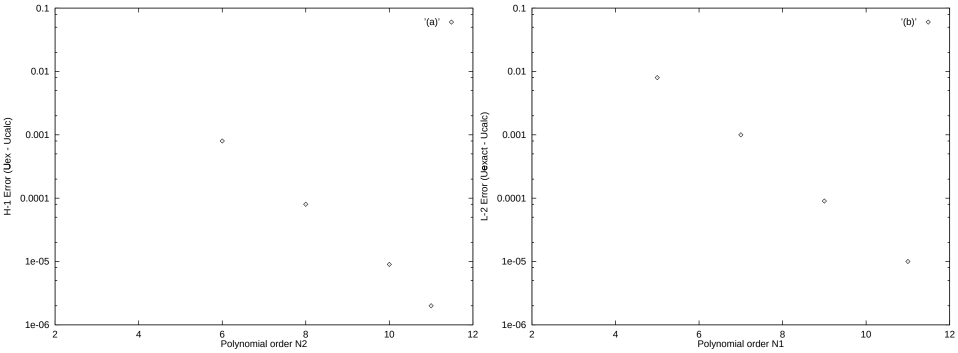
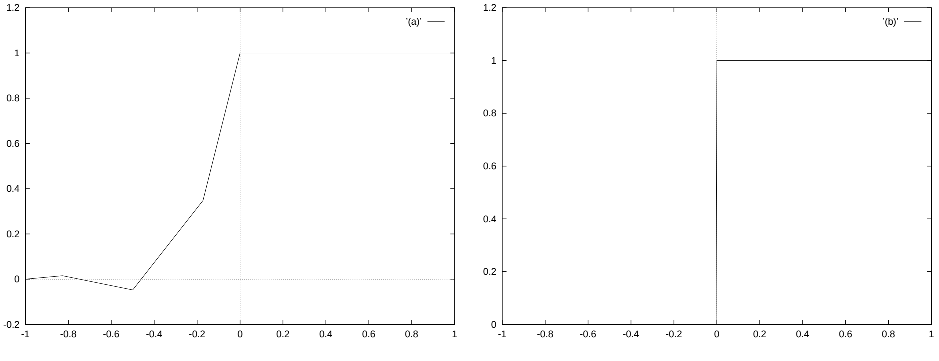


Figure 3 Case of the operator $-$ div $(\nu(.)$ **grad** $)$ with discontinuous coefficients. We consider here the numerical approximation of the solution of the Helmholtz equation $-\nu\Delta u(x, y) + u(x, y) = f(x, y)$. The domain is split into two physical subdomains: The diffusivity parameter varies from one subdomain to other; $\nu_{|\Omega_1} \gg \nu_{|\Omega_2}$: **(a)** A conforming spectral element method is used with degree polynomial $N = 5$. Obviously, one needs more refinement to has good approximation. However this objective is achieved in **(b)** with a least polynomial degree using the spectral element version of the present approach.



3 Extension to Other Cases

Heterogeneous Domain Decomposition

Heterogeneous domain decomposition have broad applications in engineering and in natural science. In this section, we give an extension of our ideas to the heterogeneous domain decomposition methods.

As an example, we consider the coupling between elliptic diffusion equations and hyperbolic convection (transport) ([AL94b],[AD96]). The idea of this procedure is that in convection-diffusion problems where the convection is dominant, the diffusion terms play a role only in the vicinity of boundary or internal layers. From the physical information, these regions can be detected a priori, it is logical to suppose that only there the complete equations have to be solved, whereas elsewhere the reduced equations can serve as a correct model. Here, we consider the following problem.

$$\begin{aligned} \beta.\ \mathbf{grad}\ u_1 + u_1 &= f_1 \text{ in } \Omega_1, \\ -\Delta u_2 + u_2 &= f_2 \text{ in } \Omega_2, \\ u_1 &= u_2, \text{ on } \Sigma, \\ \frac{\partial u_2}{\partial n_2} + \beta.n_1 u_1 &= 0 \text{ on } \Sigma, \\ u_1 &= 0 \text{ on } \Gamma_1^-, \\ u_2 &= 0 \text{ on } \Gamma_2, \end{aligned} \tag{6}$$

where

$$\Gamma_1^- = \{x \in \Gamma_1; \beta.n_1 < 0\}$$

and

$$\beta \in W^{1,+\infty}(\Omega).$$

we assume that

$$\beta(x).n_1(x) < 0 \quad \text{a.e} \quad x \in \Sigma \quad , \quad div\beta = 0.$$

Using similar arguments as in [AL94b], we can state that this problem has a unique solution.

First, as in the elliptic case, we transform (6) into an equivalent problem

$$\begin{aligned} \beta.\ \mathbf{grad}\ u_1 + u_1 &= f_1 \text{ in } \Omega_1, \\ p_2 &= \ \mathbf{grad}\ u_2 \text{ in } \Omega_2, \\ -divp_2 + u_2 &= f_2 \text{ in } \Omega_2, \\ u_1 &= u_2 \text{ on } \Sigma, \\ p_2.n_2 + \beta.n_2 u_1 &= 0 \text{ on } \Sigma, \\ u_1 &= 0 \text{ on } \Gamma_1^-, \\ u_2 &= 0 \text{ on } \Gamma_2 \end{aligned} \tag{7}$$

Let us now set $\mathcal{T}_h$ a regular triangulation of the domain Ω_1 with triangular ($d = 2$) or tetrahedral ($d = 3$) finite elements whose diameters are less or equal to h, and let k and N be positive integers. We define the finite dimensional spaces V_h and V_N by

$$V_h = \{v_h \in \mathcal{C}^0(\overline{\Omega}_1); \forall T \in \mathcal{T}_h, v_{h|T} \in P_k(T)\},$$

and

$$V_N = RT_N(\Omega_2).$$

The discrete problem is now

$$\begin{aligned} &\text{Find } (u_h, p_N) \in V_h \times V_N, \text{ such that} \\ \forall q_N \in V_N,\ & \int_{\Omega_2} (p_N.q_N + div p_N\ div q_N)dx - < q_N.n_2, u_h >_{0,\Sigma} = -\int_{\Omega_2} f\ div q_N dx, \\ \forall v_h \in V_h,\ & \sum_{T \in_h} \int_T (\beta.\ \mathbf{grad}\ u_h + u_h)(v_h + \delta h \beta.\ \mathbf{grad}\ v_h)dx + \int_\Sigma |\beta.n_1| u_h v_h d\sigma \\ & + \int_\Sigma p_N.n_2 v_h d\sigma + \int_{\Gamma_1^-} |\beta.n_1| u_h v_h d\sigma = \sum_{T \in \mathcal{T}_h} \int_T (h\beta.\ \mathbf{grad}\ v_h + v_h) f dx. \end{aligned} \tag{8}$$

where δ is a stabilization parameter.

We have the following

Theorem 3.1 *The problem (8) has a unique solution $(u_h, p_N) \in V_h \times V_N$. Moreover if the solution (u_1, p_2) of problem (6) is such that, p_2 and $div p_2$ belong to $(H^{\sigma_2}(\Omega_2))^d$ and $H^{\sigma_2}(\Omega_2)$, respectively, for a real number $\sigma_2 > 0$, and u_1 belongs to $H^{\sigma_1}(\Omega_1)$ for a real number σ_1, $1 < \sigma_1 \leq k+1$, then the following estimate holds*

$$\begin{aligned} \|p_2 - p_N\|_{H(div,\Omega_2)} & + & \|u_1 - u_h\|_{0,\Omega_1} + h^{\frac{1}{2}} \|\beta\ \mathbf{grad}\ (u_1 - u_h)\|_{0,\Omega_1} \leq \\ & C_\epsilon & \{N^{-\sigma_2+\epsilon}(\|p_2\|_{\sigma_2,\Omega_2} + \|div p_2\|_{\sigma_2,\Omega_2}) + h^{\sigma_1+\frac{1}{2}} \|u_1\|_{\sigma_1,\Omega_1}\}. \end{aligned}$$

for all $\epsilon > 0$.

Partial Differential Equations in Nonstationary Invariant Geometries

The so-called sliding schemes have been already presented in either a finite difference, [Gil88, Rai87], or mortar element framework [Ana91]. Sample candidate applications include rotating machinery and turbomachinery.

For the sake of simplicity, the method is presented for the following model problem:

$$\left\{ \begin{array}{rcll} \dfrac{\partial u}{\partial t} - \Delta\, u & = & f & \text{in } \Omega \times]0, T[\\ u(., t = 0) & = & u_0 & \text{in } \Omega \end{array} \right.$$

where $\Omega = \Omega(t)$ is a nonstationary domain, f is a given force that may depend on time, and u_0 is a given initial condition. It is obvious that we are not interested here in numerical simulation of physical situation, in that problem (3) does not take into account the equation of motion of the fluid medium. Our intent is to present the formulation of sliding interfaces problem that couples primal and equilibrium variables. We shall also focus our presentation on the simple case where $\Omega(t)$ is decomposed into two subdomains, one sliding with respect to the other along an interface $\Gamma(t)$:

$$\begin{aligned} \Omega(t) &= \Omega_1 \cup \Omega_2(t), \\ \Gamma(t) &= \overline{\Omega}_1 \cap \overline{\Omega}_2(t), \end{aligned}$$

Figure 4 One dimensional example: The domain $\Omega = (-1., 1.)^2$ is split into two physical subdomains. A spectral method is used to discretize the elliptic equation on $\Omega_1 = (-1., 0.)$, while a stabilized finite element method is used for the hyperbolic equation on $\Omega_2 = (0., 1.)$. β is constant and the right-hand side f_2 on Ω_2 is piecewise-constant. **(a)** The solution is discontinuous on the interface, and the non physical oscillations on the hyperbolic domain do not affect the elliptic domain. The continuity of the fluxes could also be illustrated. **(b)** A best approximation can be recovered on the hyperbolic solution using an adaptive finite element method based on a posteriori error estimates established for this one-dimensional problem.

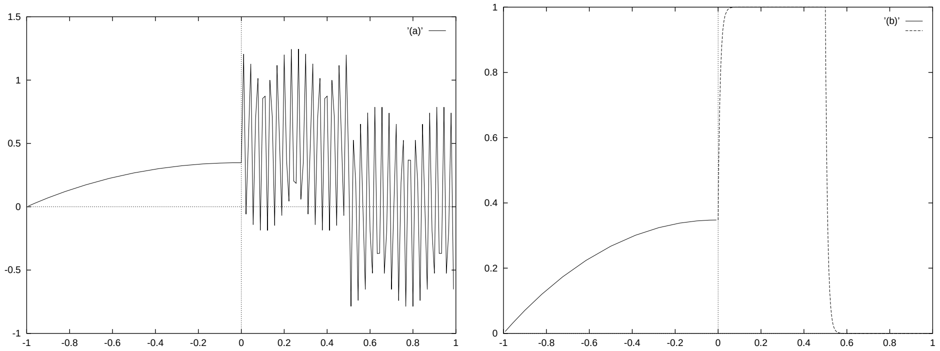
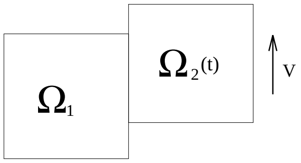


Figure 5



as illustrated in Figure 5, where $\Gamma(t)$ is a segment.

The basic formulation is of spectral element type, but the methodology we introduce is appropriate to finite elements as well. We just point out the fact that since no matching conditions are imposed on the meshes, in case of complicated geometry, one does not have to exhibit $\mathcal{C}^\infty$ mappings to use isoparametric elements. This method could then be a tool for analysing fluid flows in truly complex moving geometries, where the moving interfaces are in general curvilinear.

The scheme presented here is locally conservative, and the aliasing errors induced by numerical quadrature have no effect on the stability. This remark is mainly of interest for the implementation issue.

This variational method also preserves element-based locality, and the flexibility is evident in the treatment of mesh refinement or moving boundary problems by sliding meshes that do not introduce any mesh distortion or expensive interpolation.

Let us denote by V the velocity of $\Omega_2(t)$ and suppose it constant in time. We introduce the Lagrangian variable X in $\Omega_2(t)$ by

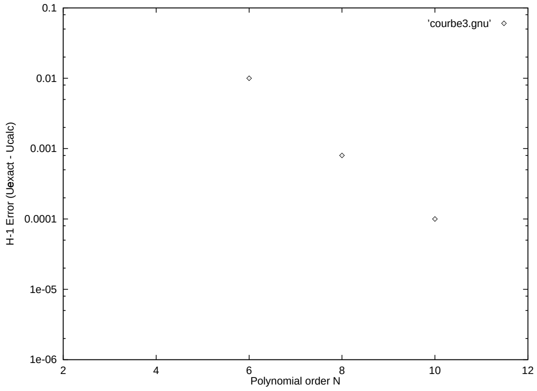
$$\left\{\begin{array}{rcl} \dfrac{\partial X}{\partial t}(x,t;\tau) & = & V(X(x,t;\tau) \\ X(x,t;\tau) & = & x. \end{array}\right.$$

Problem (3) is then expressed as,

$$\left\{\begin{array}{ll} \forall x \in \Omega_2(t), & \dfrac{\partial[u(X(x,t;\tau),t)]}{\partial t} - \Delta u(X(x,t;\tau),t) - V.\nabla u(X(x,t;\tau),t) = f(x,t), \\ \forall x \in \Omega_1, & \dfrac{\partial u}{\partial t}(x,t) \quad - \quad \Delta u(x,t) = f(x,t). \end{array}\right.$$

In practice, we can also perform internal decompositions of Ω_1 and $\Omega_2(t)$. We can choose any temporal discretization. For sake of simplicity, we deal here with a simple implicit scheme for the treatment of diffusion term, and an explicit (for example Adams-Bashforth) for the convection term. With the superscript ^{n} referring to the time $t^n = n\,\Delta t$, and u^n denoting $u(t^n,.)$, the semi-discrete problem states now as

Figure 6 Plot of discretization error $\|u - u_h\|_{H^1}$ as a function of polynomial order for the diffusion equation on the domain given by Figure 5. The exact solution is given by $u(x,y,t) = exp(-2\pi^2 t)\sin(\pi)\sin(\pi y)$. The simulation is carried out to a final time $T_f = .05$, with Δt insuring stability.



$$\left\{\begin{array}{rcl} \forall x \in \Omega_2(t^{n+1}), \quad u^{n+1} - \Delta u^{n+1} & = & \Delta t\, f^{n+1} + u^{n+1} + \Delta t(V.\nabla u^n), \\ \forall x \in \Omega_1, \quad u^{n+1} - \Delta u^{n+1} & = & \Delta t\, f^{n+1} + u^n. \end{array}\right.$$

The functional framework introduced in the previous section completes the discretization.

The proposed scheme for the approximation of problem (3) in the case of a first order time discretization reads now as follows

Find $p_{N_1}^{n+1} \in RT_{N_1}(\Omega_1)$, $u_{N_2}^{n+1} \in Q_{N_2}(\Omega_2^{n+1})$, $u_{N_2}^{n+1}|_{\partial\Omega_2\setminus\Gamma^{n+1}} = 0$ such that $\forall q^{n+1} \in RT_{N_1}(\Omega_1)$,

$$\int_{\Omega_1} (p_{N_1}^{n+1}\, q^{n+1} + \Delta t \nabla . p_{N_1}^{n+1}\, \nabla . q^{n+1})\, dx - \Delta t \int_{\Gamma^{n+1}} q^{n+1}.n\, u_{N_2}^{n+1} d\Gamma =$$
$$-\int_{\Omega_1} (\Delta t . f^{n+1} + u_{N_1}^{n})\, \nabla . q^{n+1}\, dx$$

$\forall v^{n+1} \in Q_{N_2}(\Omega_2^{n+1})$, $v^{n+1}|_{\partial\Omega_2\setminus\Gamma^{n+1}} = 0$,

$$\Delta t \int_{\Omega_2} \nabla u_{N_2}^{n+1} . \nabla v^{n+1}\, dx + \int_{\Omega_2} u_{N_2}^{n+1}\, v^{n+1}\, dx + \Delta t \int_{\Gamma^{n+1}} p_{N_1}^{n+1} .n\, v^{n+1}\, d\Gamma =$$
$$\int_{\Omega_2} (\Delta t\, f^{n+1} + u_{N_2}^{n} + \Delta t\, (V . \nabla u_{N_2}^{n})) v^{n+1}\, dx.$$

From the analysis of the previous section, we deduce that this discretization generates a unique sequence $(u_N^n)_n$ of solutions.

The analysis of the discrete problem and stability analysis of this scheme give that the error in (u, p) is bounded by a temporal error of the scheme order and spatial errors as in the Helmholtz equation. The related details of approximation results are left to a forthcoming report.

REFERENCES

[AD96] Agouzal A. and Debit N. (1996) A domain decomposition method for coupling hyperbolic and elliptic equations. Talk at 28th Congrès d'Analyse Numérique, La Londe-Les Maures.

[Ago96] Agouzal A. (1996) Méthode de décomposition de domaine en formulation mixte. *Jap. Math.* 43(1): 31–35.

[AL94a] Agouzal A. and Lamoulie L. (1994) Un algorithme de résolution pour une méthode de domaine par éléments finis. *C. R. Acad. Sci. Paris Série I* 318: 117–176.

[AL94b] Aguilar G. and Lisbona F. (1994) Interface conditions for a kind of nonlinear elliptic-hyperbolic problems. *Contemporary Mathematics* 157: 89–95.

[AMMP90] Anagnostou G., Maday Y., Mavriplis C., and Patera A. (1990) On the mortar element method: Generalization and implementation. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Proc. Third Int. Conf. on Domain Decomposition Meths.* SIAM, Philadelphia.

[Ana91] Anagnostou G. (1991) *Nonconforming sliding spectral element methods for the unsteady incompressible Navier-Stokes equations.* PhD dissertation, MIT, Cambridge, MA, Department of Mechanical Engineering.

[AT95] Agouzal A. and Thomas J. (1995) Une méthode d'éléments finis hybrides en décomposition de domaine. *M2N* 29: 749–764.

[BDM90] Bernardi C., Debit N., and Maday Y. (1990) Coupling finite element and spectral methods. *Math. Comp.* 54: 21–39.

[BMP92] Bernardi C., Maday Y., and Patera A. (1992) A new nonconforming approach to domain decomposition: The mortar element method. In *Collège de France seminar XI.* H. Brezis, J.-L. Lions.

[CHQZ88] Canuto C., Hussaini M., Quarteroni A., and Zang T. (1988) *Spectral methods in fluid dynamics.* Springer-Verlag.

[Cia91] Ciarlet P. (1991) *Basic error estimates for elliptic problems*, pages 17–352. Finite Element Methods. North-Holland, Amsterdam.

[Gil88] Giles M. (1988) Developments in the calculation of unsteady turbomachinery flow. In *Proc. Num. Meth. Fluid Dynamics III*, pages 43–63. Oxford unversity Press.

[Rai87] Rai M. (1987) Navier-stokes simulations of rotor-stator interaction using patched and overlaid grids. *J. Pow. Prop.* 3: 387–396.

[RT91] Roberts J. and Thomas J.-M. (1991) *Mixed and hybrid methods*, pages 523–639. Finite Element Methods. North-Holland, Amsterdam.

7

Classical and Cascadic Multigrid— A Methodological Comparison

Folkmar A. Bornemann and Rolf Krause

1 Introduction

We consider second order elliptic boundary value problems on a polygonal domain $\Omega \subset \mathbf{R}^d$

$$u \in H_0^1(\Omega): \qquad a(u,v) = \langle f, v\rangle_{L^2} \qquad \forall v \in H_0^1(\Omega).$$

Here $a(\cdot,\cdot)$ denotes a H_0^1-elliptic bilinear form and $f \in L^2(\Omega)$. Let

$$X_0 \subset X_1 \subset \ldots \subset X_j \subset \ldots \subset H_0^1(\Omega)$$

be a sequence of finite element spaces belonging to successively finer triangulations of Ω. On each level j the finite element solution u_j is given by a linear system

$$A_j u_j = f_j.$$

For its solution we think of an iterative scheme such that the result of m iterations with initial data u_j^0 will be denoted by $\mathcal{I}_j^m u_j^0$. The multilevel structure of the sequence $\{X_j\}$ suggests as a good idea to start the iteration on level j with the result of level $j-1$. Especially, this is of advantage in an adaptive setting where the space X_j is constructed only after a solution on level $j-1$ was obtained. Thus, on the final level ℓ an approximation u_ℓ^* of u_ℓ is computed via the multilevel scheme

$$u_0^* = u_0, \qquad u_j^* = \mathcal{I}_j^{m_j} u_{j-1}^* \qquad j = 1, \ldots, \ell. \tag{1}$$

If we choose for $\mathcal{I}_j$ the multigrid V-cycle on level j and the number of iterations as a constant m_*, i.e., $m_1 = \ldots = m_\ell = m_*$, this algorithm gives us the so-called *full multigrid* method [Hac85, Bra92]. Some authors call it *nested multigrid*.

Bornemann and Deuflhard [BD96a] considered standard iterative schemes like the damped Jacobi, the Gauss-Seidel, and the conjugate gradient iteration for $\mathcal{I}_j$. They showed that a proper choice of the number of iterations m_j on each level could make this method an "optimal" device—theoretically and practically. They named it the *cascadic multigrid method*. Its history can be obtained by browsing through

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

[Deu94, Sha96, BD96a, BD96b]. A distinctive feature of the cascadic multigrid method is the total absence of coarse grid corrections which means that coarse grids can be completely forgotten once they are refined. Therefore, the method is algorithmically attractive when a given finite element program cannot provide tree data structures of the refinement history but is used with a pre- and postprocessing device.

To be specific, the multilevel iteration (1) is called *optimal* with respect to an error norm $\|\cdot\|$ if we can choose for each final level ℓ a sequence of numbers of iterations $m_1, \ldots, m_\ell$ such that simultaneously

- the method is *accurate*, i.e., the algebraic error on level ℓ is a fraction θ below the error of discretization,
$$\|u_\ell - u_\ell^*\| \leq \theta \|u - u_\ell\|,$$
where θ is a user given constant,
- the method has *multigrid complexity*, i.e., the total computational work for the iteration is bounded by
$$\text{work } \leq c \cdot n_\ell, \tag{2}$$
where c is some constant independent of ℓ, and n_ℓ denotes the number of unknowns on level ℓ.

The full multigrid method is well known to be optimal with respect to the energy norm and the L^2-norm [Hac85, Bra92]. Bornemann and Deuflhard [BD96a, BD96b] proved optimality of the cascadic multigrid method with respect to the *energy norm.* However, optimality with respect to the L^2-norm remained an open question. For *linear* finite elements simple numerical experiments suggested that the answer to that question is negative. In this paper we will give a theoretical explanation of this fact and show that the case might be different for *higher order* elements. This will be done by a careful analysis of the two grid variant of the cascadic multigrid method. Moreover we will provide a setting where one can understand the methodological difference between the cascadic multigrid method and the classical multigrid V-cycle almost immediately. As a rule of thumb we will establish that whenever the cascadic multigrid works the classical multigrid will work too, but not vice versa.

2 Two-grid Methods — the Abstract Setting

In the Galerkin framework, a typical setting for two grid methods is given by two finite dimensional spaces
$$X_{2h} \subset X_h$$
parametrized by some discretization parameter h. These spaces should be provided for at least a sequence of parameters h converging to zero. They itself are subspaces of certain function spaces which measure *smoothness*. In the following we have to compare two *different* measures of smoothness, given by the spaces X_+, X_-,
$$X_h \subset X_+ \hookrightarrow X_-.$$

The Galerkin method is given by a projection

$$P_h : X_+ \to X_h,$$

which obeys an approximation property, or *Jackson inequality*,[1]

$$\|u - P_h u\|_{X_-} \le ch^{\sigma} \|u\|_{X_+} \qquad \forall u \in X_+.$$

Here, $\sigma > 0$ denotes some positive constant. The Galerkin method is called *optimal* if this property is complemented by an inverse inequality, or *Bernstein inequality*,

$$\|u_h\|_{X_+} \le ch^{-\sigma} \|u_h\|_{X_-} \qquad \forall u_h \in X_h.$$

Notice that the Jackson inequality implies [Bra92] the compactness of the embedding $X_+ \hookrightarrow X_-$ thus making the two measures of smoothness really different and therefore showing the necessity of introducing them. The Bernstein inequality implies that the order σ of convergence as stated in the Jackson inequality is the best possible one [Bor94, Bra92].

Moreover, the two measures of smoothness allow an abstraction of the notion of low and high frequency in the space X_h. A function $v_h \in X_h$ is of high frequency if the X_+-norm gives much larger values than the X_--norm, i.e., taking the Bernstein inequality into account,

$$\|v_h\|_{X_+} \approx h^{-\sigma} \|v_h\|_{X_-}.$$

It is of low frequency if both norms give roughly the same value,

$$\|v_h\|_{X_+} \approx \|v_h\|_{X_-}.$$

Using this abstract notion of frequency one can easily understand and formulate the properties of basic iterative schemes for the solution of the computational problem,

$$u_h = P_h u,$$

which constitutes a large, badly conditioned linear system. The error propagation after m steps of such an iterative scheme,

$$u_h - u_h^m = S^m (u_h - u_h^0),$$

will be characterized by the *smoothing property* [BD96a, BD96b]

$$\|S^m v_h\|_{X_+} \le \frac{ch^{-\sigma}}{\phi(m)} \|v_h\|_{X_-} \qquad \forall v_h \in X_h.$$

Here, we suppose $\phi(m) \to \infty$ for $m \to \infty$. With respect to the X_+-norm such a smoothing iteration reduces high frequency errors with a h-independent rate whereas low frequency error components are handled increasingly less efficient for smaller and smaller h.

[1] Here and in what follows we denote by c a generic constant independent of h.

Both the classical and the cascadic two grid method are designed to handle low frequency errors in a better way. For a given initial value $u_h^0 \in X_h$ the *classical* two grid method first performs m smoothing iterations,

$$u_h - u_h^m = S^m(u_h - u_h^0)$$

followed by a *coarse grid correction*,

$$u_h^* = u_h^m + P_{2h}(u_h - u_h^m).$$

The combined error of these two steps is given by

$$\begin{aligned} \|u_h - u_h^*\|_{X_-} &= \|(u_h - u_h^m) - P_{2h}(u_h - u_h^m)\|_{X_-} \leq ch^\sigma \|u_h - u_h^m\|_{X_+} \\ &\leq \frac{c}{\phi(m)} \|u_h - u_h^0\|_{X_-}, \end{aligned}$$

where we have used first the Jackson inequality and second the smoothing property. Thus, we end up with an error reduction in the X_--norm which is independent of the discretization parameter h.

The *cascadic* two grid method changes somewhat the order of the two steps of the classical method. It first performs a coarse grid projection

$$u_h^0 = P_{2h} u_h$$

followed by m smoothing iterations,

$$u_h - u_h^* = S^m(u_h - u_h^0).$$

Here the combined error is given by

$$\begin{aligned} \|u_h - u_h^*\|_{X_+} &\leq \frac{ch^{-\sigma}}{\phi(m)} \|u_h - P_{2h} u_h\|_{X_-} = \frac{ch^{-\sigma}}{\phi(m)} \|(u_h - u_h^0) - P_{2h}(u_h - u_h^0)\|_{X_-} \\ &\leq \frac{c}{\phi(m)} \|u_h - u_h^0\|_{X_+}, \end{aligned}$$

using first the smoothing property and then the Jackson inequality. Here we end up with an error reduction in the X_+-norm which is independent of the discretization parameter h.

Notice, however, some important differences between the two methods. Since the order of the smoothing iterations and the coarse grid problem are interchanged, and therefore the Jackson inequality and the smoothing property (which resembles the Bernstein inequality) are applied in reverse order, the error reduction occurs in *different norms*: For the classical two grid in X_-, for the cascadic two grid in X_+. As we will see later on this is the reason for essentially different behavior in certain settings. A second methodological difference is of algorithmic nature: Unlike the classical two grid method, the cascadic two grid method has no choice of an initial value, which precludes it from being part of an iterative scheme itself.

3 Application to Finite Element Spaces

Here, we apply the abstract theory of the preceding section to k-th order finite elements. We consider two choices for the norm in which error reduction of the iterative scheme is measured: energy norm and L^2-norm. These two examples clearly reveal the general principle.

Energy Norm and Classical Multigrid

Here, error reduction is obtained in the X_--norm and we thus put $X_- = H^1(\Omega)$. The more smooth space is set to $X_+ = H^{1+\alpha}(\Omega)$ where $\alpha > 0$ is chosen such that the corresponding Jackson and Bernstein inequalities hold:

$$\|u - P_h u\|_{H^1} \le ch^\alpha \|u\|_{H^{1+\alpha}}, \qquad \|u_h\|_{H^{1+\alpha}} \le ch^{-\alpha}\|u_h\|_{H^1} \tag{3}$$

for all $u \in H^{1+\alpha}$ and $u_h \in X_h$. The smoothness parameter[2] α is restricted by the regularity of the elliptic problem and by the order of the chosen finite elements: For $H^{1+\mu}$-regular elliptic problems we can take any α as large as

$$\alpha \le \min(\mu, k).$$

For $\alpha \ge 1/2$, the term $\|u_h\|_{H^{1+\alpha}}$ is meant to denote the corresponding *discrete* Sobolev norm [Bra92]. Now, our abstract theory yields the h-independent error reduction

$$\|u_h - u_h^*\|_{H^1} \le \frac{c}{\phi(m)}\|u_h - u_h^0\|_{H^1}.$$

Energy Norm and Cascadic Multigrid

Here, error reduction is obtain in the X_+-norm and we thus put $X_+ = H^1(\Omega)$. Whenever the Jackson and Bernstein inequalities (3) of the $(H^1, H^{1+\alpha})$ pair holds we get by a Aubin-Nitsche type of duality argument corresponding Jackson- and Bernstein inequalities for the pair $(H^{1-\alpha}, H^1)$,

$$\|u - P_h u\|_{H^{1-\alpha}} \le ch^\alpha \|u\|_{H^1}, \qquad \|u_h\|_{H^1} \le ch^{-\alpha}\|u_h\|_{H^{1-\alpha}} \tag{4}$$

for all $u \in H^1$ and $u_h \in X_h$. As in the case of the classical multigrid method we thus get the h-independent error reduction

$$\|u_h - u_h^*\|_{H^1} \le \frac{c}{\phi(m)}\|u_h - u_h^0\|_{H^1}.$$

L^2-Norm and Classical Multigrid

Here, we put $X_- = L^2(\Omega)$ and assume enough regularity for $\alpha = 1$ in (3) and (4). Taking $X_+ = H^1(\Omega)$ the Jackson and Bernstein inequalities (4) lead to the h-independent convergence rate

$$\|u_h - u_h^*\|_{L^2} \le \frac{c}{\phi(m)}\|u_h - u_h^0\|_{L^2}.$$

[2] For a polygonal domain Ω and piecewise smooth coefficients of the elliptic operator we always get some $\alpha > 0$.

L^2-Norm and Cascadic Multigrid

We have to put $X_+ = L^2(\Omega)$. Thus, X_- must be a Sobolev space of *negative* order, say $X_- = H^{-\epsilon}(\Omega)$, $\epsilon > 0$. Our argument would work, if we had a Jackson inequality like

$$\|u - P_h u\|_{H^{-\epsilon}} \leq ch^{\epsilon}\|u\|_{L^2}. \tag{5}$$

Assuming the same regularity as for the classical two grid method, i.e., at least $\alpha = 1$, and replacing u by $u - P_h u$ in (5) we get

$$\|u - P_h u\|_{H^{-\epsilon}} \leq ch^{\epsilon}\|u - P_h u\|_{L^2} \leq ch^{1+\epsilon}\|u - P_h u\|_{H^1}.$$

However, for a right hand side $f \in H^{\epsilon}(\Omega)$ this would be possible *only if* we could impose enough regularity: By duality

$$\|u - P_h u\|_{H^{-\epsilon}} \geq \frac{\langle u - P_h u, f\rangle_{L^2}}{\|f\|_{H^{\epsilon}}} = \frac{a(u - P_h u, u - P_h u)}{\|f\|_{H^{\epsilon}}} \geq c\,\frac{\|u - P_h u\|_{H^1}^2}{\|f\|_{H^{\epsilon}}}$$

we would get the H^1-estimate

$$\|u - P_h u\|_{H^1} \leq ch^{1+\epsilon}\,\|f\|_{H^{\epsilon}},$$

which means that $\alpha = 1 + \epsilon$ would be admissible, cf. [Bor94].

In particular, the hypothetical Jackson inequality (5) does *not* hold for *linear* finite elements, $k = 1$, where we are restricted to $\alpha \leq 1$. In this case the only estimate we can prove for the cascadic two grid method is

$$\|u_h - u_h^*\|_{L^2} \leq \frac{c_\epsilon h^{-\epsilon}}{\phi_\epsilon(m)}\|u_h - u_h^0\|_{L^2}.$$

Using a damped Jacobi or Gauss-Seidel iteration, we have $\phi_\epsilon(m) = m^{\epsilon/2}$ as shown in [BD96a]. *Assuming* that our estimate is essentially a sharp one we would expect that the number of iterations m_h, which is needed to reduce the algebraic error a given amount, *increases* like

$$m_h \propto h^{-2}.$$

This was observed in several numerical experiments. Hence our estimates appear to be rather sharp and there is not much to improve.

Discussion

The Aubin-Nitsche duality argument and the reverse duality argument of the last paragraph show that Jackson and Bernstein inequalities for the finite element spaces are located exactly in the smoothness range between $H^{1-\alpha}$ and $H^{1+\alpha}$ which is *symmetric* with respect to H^1. For this reason the classical and the cascadic two grid method have the same chance to locate the partner space $X_\pm$ of the space $X_\mp$ that measures the energy norm. However, since the space L^2 is located in this smoothness range in a way leaving more place for more smooth spaces than for less smooth spaces it gives preference to the *classical* two grid method which puts $X_- = L^2$ and needs a more smooth partner space X_+.

4 Remarks on Optimality in the L^2-NORM

As we have seen, the cascadic two grid method works for the L^2-norm if we have $\alpha > 1$. *Assuming* this regularity we will discuss shortly whether we can prove optimality of the cascadic multigrid method with respect to L^2 on uniform triangulations. We follow quite closely the proof given in [BD96a, BD96b] for the energy norm case.

By linearity the basic error estimate governing the multilevel iteration (1) is now given by

$$\|u_j - u_j^*\|_{L^2} \le \|S^{m_j}(u_j - u_{j-1})\|_{L^2} + \|S^{m_j}(u_{j-1} - u_{j-1}^*)\|_{L^2}.$$

Applying that recursively we get by setting $M_j = m_\ell + \ldots + m_j$ the estimate

$$\begin{aligned} \|u_\ell - u_\ell^*\|_{L^2} &\le \sum_{j=1}^{\ell} \|S^{M_j}(u_j - u_{j-1})\|_{L^2} \le c \sum_{j=1}^{\ell} \frac{h_j^{1-\alpha}}{M_j^{\gamma(\alpha-1)}} \|u_j - u_{j-1}\|_{H^{1-\alpha}} \\ &\le c \sum_{j=1}^{\ell} \frac{1}{M_j^{\gamma(\alpha-1)}} \|u_j - u_{j-1}\|_{L^2}, \end{aligned}$$

where we have used the smoothing property and the Jackson inequality (5) with $\epsilon = \alpha - 1$. Moreover we specified the function ϕ of the smoothing property by $\phi(m) = m^{\epsilon\gamma}$, where $\gamma = 1/2$ for the damped Jacobi or Gauss-Seidel iteration and $\gamma = 1$ for the CG-iteration [BD96a, BD96b]. As discussed in these references we set

$$m_j = \lceil \beta^{\ell - j} m_\ell \rceil.$$

Taking into account the L^2-error estimate

$$\|u_j - u_{j-1}\|_{L^2} \le c h_j^{1+\alpha} \|u\|_{H^{1+\alpha}},$$

we finally get the estimate

$$\|u_\ell - u_\ell^*\|_{L^2} \le \frac{c h_\ell^{1+\alpha}}{m_\ell^{\gamma(\alpha-1)}} \sum_{j=0}^{\ell-1} \left(\frac{2^{\alpha+1}}{\beta^{\gamma(\alpha-1)}} \right)^j \|u\|_{H^{1+\alpha}}.$$

Thus, *accuracy* of the method is guaranteed if the sum can be bounded independently of the final level ℓ. This is the case if and only if

$$\beta > 2^{\frac{1}{\gamma} \frac{\alpha+1}{\alpha-1}}.$$

As shown in [BD96a, BD96b] multigrid complexity is obtained if and only if $\beta < 2^d$, where d is the dimension of the domain Ω. Thus, a sufficient condition for the cascadic multigrid to be optimal with respect to the L^2-norm is

$$d > \frac{1}{\gamma} \cdot \frac{\alpha + 1}{\alpha - 1}. \tag{6}$$

If we relax the demand (2) for multigrid complexity to

$$\text{work} \le c \cdot n_\ell \log^p n_\ell, \tag{7}$$

for some $p > 0$, one can show exactly in the same way as in [BD96a] for the energy norm that equality is admissible in condition (6).

Examples

This restrictive condition will be illuminated by several specific cases:

- $d = 2$, $\gamma = 1$ (CG-iteration): Condition (6) is equivalent to $\alpha > 3$, which means at least H^σ-regularity with $\sigma > 4$ and order $k > 3$ finite elements.
- $d = 3$, $\gamma = 1$ (CG-iteration): Condition (6) is equivalent to $\alpha > 2$, which means at least H^σ-regularity with $\sigma > 3$ and order $k > 2$ finite elements.
- $d = 2$, $\gamma = 1/2$ (damped Jacobi or Gauss-Seidel iteration): Condition (6) is *not* satisfied for any $\alpha > 1$. In this case the cascadic multigrid method is not optimal with respect to the L^2-norm for any regularity and any order of finite elements!
- $d = 3$, $\gamma = 1/2$ (damped Jacobi or Gauss-Seidel iteration): Condition (6) is equivalent to $\alpha > 5$ which means at least H^σ-regularity with $\sigma > 6$ and order $k > 5$ finite elements.

For the relaxed multigrid complexity (7) equality is admissible in all cases.

REFERENCES

[BD96a] Bornemann F. A. and Deuflhard P. (1996) The cascadic multigrid method for elliptic problems. *Numer. Math.* 75: 135–152.

[BD96b] Bornemann F. A. and Deuflhard P. (1996) Cascadic Multigrid Methods. In Glowinski R., Périaux J., Shi Z., and Widlund O. (eds) *Domain Decomposition Methods in Sciences and Engineering*. John Wiley & Sons, Chichester, New York.

[Bor94] Bornemann F. A. (1994) Interpolation spaces and optimal multilevel preconditioners. In Keyes D. and Xu J. (eds) *Proceedings of the 7th International Conference on Domain Decomposition Methods 1993*, pages 3–8. AMS, Providence.

[Bra92] Braess D. (1992) *Finite Elemente*. Springer-Verlag, Berlin, Heidelberg, New York.

[Deu94] Deuflhard P. (1994) Cascadic conjugate gradient methods for elliptic partial differential equations. Algorithm and numerical results. In Keyes D. and Xu J. (eds) *Proceedings of the 7th International Conference on Domain Decomposition Methods 1993*, pages 29–42. AMS, Providence.

[Hac85] Hackbusch W. (1985) *Multi-Grid Methods and Applications*. Springer-Verlag, Berlin, Heidelberg, New York.

[Sha96] Shaidurov V. V. (1996) Some estimates of the rate of convergence for the cascadic conjugate-gradient method. *Computers Math. Applic.* 31: 161–171.

8

Schwarz Preconditioners for the Spectral Element Stokes and Navier-Stokes Discretizations

Mario A. Casarin

1 Introduction

We consider fast methods of solving the linear system

$$\left\{ \begin{array}{l} A\underline{\mathbf{u}} + B^t\underline{p} = \underline{\mathbf{f}} \\ \\ \quad B\underline{\mathbf{u}} = \underline{0}. \end{array} \right. \tag{1}$$

resulting from the discretization of the Stokes problem by the spectral element method; see (3).

The efficient solution of this and analogous systems, generated by a variety of discretization methods, has been the object of various studies. The Uzawa procedure is a relatively standard technique [GPAR86], and more recently block-diagonal and block-triangular preconditioners have been proposed [Elm94, Kla97]. Global pressure variables are used in [BP89] and [TP95] as Lagrange multipliers to constrain the interface velocities and to guarantee that the divergence free condition holds.

Rønquist has proposed an iterative substructuring method that is based on a decomposition of the domain into interiors of subregions, faces, edges, and vertices. The coarse problem is a Stokes problem approximated by a lower-dimensional pair of discrete spaces on the coarse mesh. Stokes problems are solved within the subregions, while a diagonal scaling using elements of the matrix A is performed on the interface velocity variables. This scheme avoids costly inner iterations, and its built-in parallelism is certainly a very desirable feature. In [Røn95], relatively large problems in three dimensions are solved with modest computer resources. The small iteration count and the excellent approximation properties of the spectral element method for flow problems makes this a very efficient scheme.

Inspired by Rønquist's scheme, we have developed iterative substructuring methods, for which the velocities are restricted to the space of discretely divergence-free functions in the spectral element sense. The PCG method is applied to the resulting symmetric, positive definite linear system. The condition number of our algorithms grows at most like

$$\frac{C(1+\log(N))^3}{\beta_N},$$

where β_N is the Babuška-Brezzi constant; see Lemma 2.1. Our approach is also related to the methods of [BP89] and [TP95].

The next section introduces the details of the discretization method. Section 3 presents an important extension operator, while in Section 4, the $Q_2 - Q_0$ pair is used to generate a coarse space for the Stokes problem. The theory carries over without any substantial change to a variety of mixed discretizations using a discontinuous pressure space.

In Sections 5 and 6, we extend the Schwarz theory for indefinite and non-symmetric problems to the Navier-Stokes problem, taking advantage of the Stokes preconditioner developed here. In each step of Newton's method, only the velocity of the previous step is used. The pressure is computed only when required, typically after the velocity has been obtained to the prescribed accuracy. The key point in the success of this method is the construction of an appropriate coarse space.

2 Discretization Method

Let Ω be a domain in $I\!R^d$, $d = 2$ or 3. We triangulate Ω into non-overlapping *substructures* $\{\Omega_i\}_{i=1}^M$ of diameter H_i. Each Ω_i is the image of the reference substructure $\hat{\Omega} = [-1,+1]^3$ under a mapping $F_i = D_i \circ G_i$ where D_i is an isotropic dilation and G_i a C^∞ mapping such that its Jacobian and the inverse thereof are uniformly bounded by a constant. We assume, e.g., in three dimensions, that the intersection between the closures of two distinct substructures is either empty, a vertex, a whole edge or a whole face.

We define the space $P^N(\hat{\Omega})$ as the space of polynomials of degree at most N in each of the variables separately. The space $P^N(\Omega_i)$ is the space of functions v_N such that $v_N \circ F_i$ belongs to $P^N(\hat{\Omega})$. The conforming discretization space $P_0^N(\Omega) \subset H_0^1(\Omega)$ is the space of continuous functions the restrictions of which to Ω_i belong to $P^N(\Omega_i)$.

Let $\Lambda = [-1,1]$. For each N, the Gauss-Lobatto-Legendre quadrature of order N is denoted by GLL(N) and satisfies: $\forall p \in P^{2N-1}(\Lambda), \quad \int_{-1}^{1} p(x)\,dx \;= \sum_{j=0}^{N} p(\xi_j)\rho_j$. Here, the quadrature points ξ_j are numbered in increasing order, and are the zeros of $(1-x^2)L_N'(x)$, and $L_N(x)$ is the Legendre polynomial of degree n.

In three dimensions, the discrete $L^2(\Omega)$-inner product is defined by

$$(u,v)_N = \sum_{i=1}^{M} \sum_{j,k,l=0}^{N} (u \circ F_i) \cdot (v \circ F_i) \cdot |J_i|(\xi_j, \xi_k, \xi_l) \cdot \rho_j \rho_k \rho_l, \tag{2}$$

where $|J_i|$ is the Jacobian determinant of F_i.

We next consider the variational form of the Stokes equation in the velocity-pressure formulation, discretized by the spectral elements. While the velocities are taken to be continuous functions, the pressures can be discontinuous across substructure boundaries. The restriction of the pressure space $\bar{P}^{N-2}(\Omega)$ to each Ω_i is $P^{N-2}(\Omega_i)$. We note that $\bar{P}^{N-2}(\Omega) \subset L^2(\Omega)$, but $\bar{P}^{N-2}(\Omega) \not\subset H^1(\Omega)$.

The discrete problem is given by:
Find $(\mathbf{u}, p) \in (P_0^N(\Omega))^d \times \bar{P}^{N-2}(\Omega) \cap L_0^2(\Omega)$ such that:

$$\left\{ \begin{array}{rcl} a_Q(\mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) & = & (\mathbf{f}, \mathbf{v})_N \quad \forall \mathbf{v} \in (P_0^N(\Omega))^d, \\ \\ b(\mathbf{u}, q) & = & 0 \quad \forall q \in \bar{P}^{N-2}(\Omega) \cap L_0^2(\Omega). \end{array} \right. \tag{3}$$

Here, $a_Q(\cdot, \cdot)$ is given by $a_Q(\mathbf{u}, \mathbf{v}) = \sum_{i,j=1}^d \nu \left(\frac{\partial \mathbf{u}_i}{\partial x_j}, \frac{\partial \mathbf{v}_i}{\partial x_j} \right)_N$. We assume, for simplicity, that $b(\mathbf{v}, q) = -\int_\Omega q \nabla \cdot \mathbf{v} \, dx$; see [MPR92]. The right-hand side is assumed to be in $L^2(\Omega)$. Our analysis also applies to the more general non-homogeneous problem, and also to mixed Dirichlet and Neumann boundary conditions, with only minor changes.

For the velocities, we choose standard nodal basis functions $\phi_j^N \in (P_0^N(\Omega))^d$. We number the GLL$(N)$ nodes ξ *within* the subregions Ω_i by an index r, and define a basis for $\bar{P}^{N-2}(\Omega)$ by $\beta_{r_1}(\xi_{r_2}) = \delta_{r_1 r_2}$, for all r_1, r_2, where δ is the Kronecker symbol. We note that any function of $\bar{P}^{N-2}(\Omega)$ is uniquely represented by its values at the interior GLL(N) nodes ξ_r. By writing the system (3) in terms of these two bases, we arrive, in a standard way, at the system (1). To each component of the velocity, there corresponds a diagonal block of A which is equal to the standard scalar spectral element stiffness matrix K_N. The entries of B are given by $B_{jr} = b(\phi_j^N, \beta_r)$, and $\underline{\mathbf{f}}$ is a vector with components $\underline{\mathbf{f}}_j = (\mathbf{f}, \phi_j^N)$.

The next lemma is the key point in the error analysis of this discretization; see [MPR92].

Lemma 2.1 *For each N, there exists a $\beta_N > 0$ such that*

$$\inf_{q \in \bar{P}^{N-2}(\Omega) \cap L_0^2(\Omega)} \quad \sup_{\mathbf{v} \in (P_0^N(\Omega))^d} \frac{b(\mathbf{v}, q)}{||\mathbf{v}||_{H^1(\Omega)} ||q||_{L^2(\Omega)}} \geq \beta_N.$$

If the geometry is rectilinear, i.e. the F_i are affine mappings, then there exists a constant β, independent of N, and such that $\beta_N \geq \beta N^{\frac{1-d}{2}}$, for $d = 1$, 2, or 3.

We remark that very good convergence properties are predicted by the theory and have been extensively verified in practice; see, e.g., [FR94].

3 An Extension Operator

For a subregion Ω_i, we define an extension operator $E_i^{S,N} : (P^N(\partial\Omega_i))^3 \longrightarrow (P^N(\Omega_i))^3$, where $\mathbf{u}_i = E_i^{S,N}(\mathbf{g}_i)$ is the velocity component of the solution to the following Stokes problem:
Find $(\mathbf{u}_i, p_i) \in ((P^N(\Omega_i))^d, P^{N-2}(\Omega_i) \cap L_0^2(\Omega_i))$, such that:

$$\left\{ \begin{array}{l} a_Q(\mathbf{u}_i, \mathbf{v}_i) + b_{\Omega_i}(\mathbf{v}_i, p_i) = 0 \quad \forall \mathbf{v}_i \in (P_0^N(\Omega_i))^d, \\ b_{\Omega_i}(\mathbf{u}_i, q_i) = 0 \quad \forall q \in P^{N-2}(\Omega_i) \cap L_0^2(\Omega_i), \\ \mathbf{u}_i|_{\partial\Omega_i} = \mathbf{g}_i. \end{array} \right. \tag{4}$$

The subscript Ω_i indicates that the integration or quadrature is taken on Ω_i only. In other words, $\mathbf{u}_i$ is the solution of a homogeneous Stokes problem with $\mathbf{g}_i$ as boundary data, and zero right-hand side within Ω_i. We remark that $\mathbf{u}_i$ always exists, even if the outward fluxes $\int_{\partial\Omega_i} \mathbf{g}_i \cdot \mathbf{n}\, dS$ are not equal to zero, since the pressure test space does not include the constant function, and the Babuška-Brezzi condition is satisfied for the problems restricted to each subregion.

We remark that if $\mathbf{u}_i = E_i^{S,N}(\mathbf{g}_i)$, then

$$a_{Q,\Omega_i}(\mathbf{u}_i, \mathbf{u}_i) = \min_{\mathbf{v}_i|_{\partial\Omega_i} = \mathbf{g}_i} a_{Q,\Omega_i}(\mathbf{v}_i, \mathbf{v}_i) \quad \forall \mathbf{v}_i \in P_\nabla^N(\Omega_i), \tag{5}$$

where $P_\nabla^N(\Omega_i) = \{\mathbf{v}_i \in (P^N(\Omega_i))^d \mid \ b_{\Omega_i}(\mathbf{v}_i, q_i) = 0 \quad \forall q_i \in P^{N-2}(\Omega_i) \cap L_0^2(\Omega_i)\}$.

Let $P_{0,\nabla}^N(\Omega)$ be the space of discretely divergence-free functions i.e. functions that satisfy the second equation of (3). For $\mathbf{v} \in (P_0^N(\Omega))^d$, let $\tilde{\mathbf{v}}$ be defined by $\tilde{\mathbf{v}}|_{\Omega_i} = E_i^{S,N}(\mathbf{v}|_{\partial\Omega_i})$. It is easy to see that if $\int_{\partial\Omega_i} \mathbf{v} \cdot \mathbf{n}\, dS = 0 \ \forall i$, then $\tilde{\mathbf{v}} \in P_{0,\nabla}^N(\Omega)$.

4 A Domain Decomposition Preconditioner

We describe the construction in detail for two dimensions. The three-dimensional case is analogous; see [GPAR86], Section II.3.1, and Remark 4.1 below. For a reference square $\hat{\Omega} = [-1, 1]^2$, let

$$V_{\mathbf{n}}^H(\hat{\Omega}) = (Q_1(\hat{\Omega}))^2 \oplus \operatorname{span}\{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4\},$$

where $\mathbf{p}_i \in (Q_2(\hat{\Omega}))^2$ vanishes on the edges $\mathcal{E}_j$ for $j \neq i$, and is normal to $\mathcal{E}_i$. For example, for the edge $\mathcal{E}_1$ given by $x = 1$, $\mathbf{p}_1 = ((1 + x)(1 - y^2), 0)$.

The space $V_{\mathbf{n}}^H(\Omega) \subset (H_0^1(\Omega))^2$ is the space whose restrictions to each Ω_i is the image of $V_{\mathbf{n}}^H(\hat{\Omega})$ under the mapping F_i, which is here taken to be isoparametric with respect to the space $(Q_1(\hat{\Omega}))^2$; see [GPAR86], Section A.2. There are 12 degrees of freedom per element, namely the nodal values at each vertex and the fluxes across each of the edges.

Let $Q_0^H(\Omega)$ be the space of functions of zero mean on Ω that are constant within each substructure Ω_i. It is well-known that for the discretization of the Stokes problem on the coarse mesh, the pair $V_{\mathbf{n}}^H - Q_0^H$ yields a stable discretization in the Babuška-Brezzi sense, with a stability constant bounded away from zero independently of H.

Let $V_{\mathbf{n},\nabla_H}^H(\Omega)$ be defined by: $V_{\mathbf{n},\nabla_H}^H(\Omega) = \{\mathbf{u} \in V_{\mathbf{n}}^H \mid \int_{\Omega_i} \nabla \cdot \mathbf{u} dx = \int_{\partial\Omega_i} \mathbf{u} \cdot \mathbf{n} dS = 0\}$. This space plays the role of our coarse space, but it is clearly not contained in $P_{0,\nabla}^N(\Omega)$, since a function $\mathbf{u} \in V_{\mathbf{n},\nabla_H}^H(\Omega)$ in general fails to have a divergence orthogonal to the space $\bar{P}^{N-2}(\Omega)$ in $L^2(\Omega)$. We therefore define a transfer operator $I_H^h : V_{\mathbf{n},\nabla_H}^H(\Omega) \to P_{0,\nabla}^N(\Omega)$ by:

$$\begin{cases} I_H^h(\mathbf{u}_H)|_{\partial\Omega_i} = \mathbf{u}_H|_{\partial\Omega_i} \\ I_H^h(\mathbf{u}_H)|_{\Omega_i} = E_i^{S,N}(\mathbf{u}_H|_{\partial\Omega_i}). \end{cases} \tag{6}$$

This operator satisfies the usual H^1-stability and L^2-approximation properties used in the Schwarz theory.

For $\mathbf{u}, \mathbf{v} \in H^1(\Omega)$, we define the bilinear form $a(\mathbf{u}, \mathbf{v}) := \int_\Omega \nabla\mathbf{u} \cdot \nabla\mathbf{v}\, dx$. The coarse solver T_h^H is given by

$$a(T_h^H \mathbf{u}, \mathbf{w}) = a_Q(\mathbf{u}, I_H^h \mathbf{w}) \quad \forall \mathbf{w} \in V_{\mathbf{n},\nabla_H}^H(\Omega).$$

For each edge $\mathcal{E}_k$ shared by two subregions Ω_i and Ω_j, let Ω_{ij} be the union of Ω_i, Ω_j, and $\mathcal{E}_k$. The local space $V_{\mathcal{E}_k} \subset P^N_{0,\nabla}(\Omega)$ consists of functions $\mathbf{u}_{\mathcal{E}_k}$ with support in $\bar{\Omega}_{ij}$, and whose values in the interior of Ω_i and Ω_j are given by $E_i^{S,N}$ and $E_j^{S,N}$, respectively. This definition implies that $\forall \mathbf{u}_{\mathcal{E}_k} \in V_{\mathcal{E}_k}$, $\int_{\mathcal{E}_k} \mathbf{u}_{\mathcal{E}_k} \cdot \mathbf{n}\, dS = 0$. The bilinear form associated with $V_{\mathcal{E}_k}$ is $a_Q(\cdot,\cdot)$.

For each interior vertex v_n, let $\mathcal{E}(v_n)$ be the collection of all edges having v_n as an endpoint. We define $\phi_{v_n,x} \in P^N_{0,\nabla}(\Omega)$ by assigning values at the interface nodes, and using the $E_i^{S,N}$ to extend these values to the interior of the substructures. We let $\phi_{v_n,x}(v_n) = (1,0)$, let $\phi_{v_n,x}$ be equal to zero at all the interface nodes not adjacent to v_n, and $\forall \mathcal{E}_k \in \mathcal{E}(v_n)$, we let $\phi_{v_n,x}$ be equal to a constant vector at the node v'_n next to v_n on the edge $\mathcal{E}_k$. This constant vector is taken to be normal to the interface at v'_n, and so that $\int_{\mathcal{E}_k} \phi_{v_n,x} \cdot \mathbf{n}\, dS = 0$. We define $\phi_{v_n,y}$ analogously. The one-dimensional vertex spaces are given by:

$$V_{v_n,x} = \mathrm{span}\{\phi_{v_n,x}\} \text{ and } V_{v_n,y} = \mathrm{span}\{\phi_{v_n,y}\}.$$

The bilinear form associated with the vertex spaces is $a_Q(\cdot,\cdot)$.

The interior spaces are $V_{\Omega_i} = P^N_{0,\nabla}(\Omega_i)$, and the bilinear form associated with all of them is $a_Q(\cdot,\cdot)$.

The preconditioned operator is now

$$T_{\mathbf{n}} = I_H^h T_h^H + \sum_{v_n} (T_{v_n,x} + T_{v_n,y}) + \sum_{\mathcal{E}_k} T_{\mathcal{E}_k} + \sum_{i=1}^{M} T_{\Omega_i}. \tag{7}$$

This operator does not exactly fit the Schwarz framework, but an analyisis similar to the proof of that result, together with a decomposition lemma involving the local spaces and bilinear forms just described, yield the following theorem. For the proof, see [Cas96]; cf. [Bre94, Cai95].

Theorem 4.1 *The condition number of $T_{\mathbf{n}}$ satisfies:*

$$\kappa(T_{\mathbf{n}}) \leq \frac{C(1+\log(N))^3}{\beta_N}.$$

Remark 4.1 *In three dimensions, edge and face functions play the role of the vertex and edge functions of the two dimensional version, respectively. For each edge, the edge function is the analogue of the ϕ_{v_n} above; it is nonzero for the interface nodes adjacent to the edge, and have zero flux across all the faces of the subregions. The condition number estimate is the same as in Theorem 4.1, where β_N is now the Babuška-Brezzi constant for the three-dimensional discretization.*

5 Schwarz Methods for the Stationary Navier-Stokes Equations

Following [Røn95], we consider a Galerkin spectral element discretization of the velocity-pressure formulation of the Navier-Stokes equations, given by:

Find $(\tilde{\mathbf{u}}_N, \tilde{p}_{N-2}) \in P_0^N(\Omega) \times (\bar{P}^{N-2}(\Omega) \cap L_0^2(\Omega))$ such that

$$\left\{ \begin{array}{l} a(\tilde{\mathbf{u}}_N, \mathbf{v}_N) + c(\tilde{\mathbf{u}}_N; \tilde{\mathbf{u}}_N, \mathbf{v}_N) + b(\mathbf{v}_N, \tilde{p}_{N-2}) = \int_\Omega \mathbf{f} \cdot \mathbf{v}_N \, dx \quad \forall \mathbf{v}_N \in P_0^N(\Omega), \\ \qquad b(\tilde{\mathbf{u}}_N, q_{N-2}) = 0 \quad \forall q_{N-2} \in \bar{P}^{N-2}(\Omega) \cap L_0^2(\Omega). \end{array} \right. \tag{8}$$

For $\mathbf{u}, \mathbf{v}$, and $\mathbf{w} \in H^1(\Omega)$, the trilinear form $c(\cdot; \cdot, \cdot)$ is given by:

$$c(\mathbf{u}; \mathbf{v}, \mathbf{w}) := \sum_{i,j=1}^{d} \int_\Omega \mathbf{u}_j (\frac{\partial \mathbf{v}_i}{\partial x_j}) \mathbf{w}_i \, dx.$$

Numerical computations show that $\tilde{\mathbf{u}}_N$ is a good approximation for $\mathbf{u}$, the exact solution of the Navier-Stokes equations, at least for Reynolds number $Re = 1/\nu$ on the order of 50; see [Røn95].

We will develop Schwarz preconditioners for the system representing the k^{th} step of the Newton iteration used to solve (8). We fix k, and to simplify notations, set $\mathbf{u}_N := \mathbf{u}_N^k$, $\mathbf{w} := \mathbf{u}_N^{k-1}$, and $\mathbf{g} := \mathbf{f}^k$. Then, $\mathbf{u}_N$ is the solution of the following problem:
Find $\mathbf{u}_N \in P_{0,\nabla}^N(\Omega)$ such that

$$B_{\mathbf{w}}(\mathbf{u}_N, \mathbf{v}_N) = (\mathbf{g}, \mathbf{v}_N) \quad \mathbf{v}_N \in P_{0,\nabla}^N(\Omega), \tag{9}$$

where

$$B_{\mathbf{w}}(\mathbf{u}_N, \mathbf{v}_N) = a(\mathbf{u}_N, \mathbf{v}_N) + c(\mathbf{w}; \mathbf{u}_N, \mathbf{v}_N) + c(\mathbf{u}_N; \mathbf{w}, \mathbf{v}_N). \tag{10}$$

We assume that $\tilde{\mathbf{u}}_N$ is a solution of (8) which is non-singular i.e. (9) is uniquely solvable if we let $\mathbf{w} = \tilde{\mathbf{u}}_N$. If the Reynolds number Re $= 1/\nu$ is small enough, this can be proved by classical arguments (see [GPAR86], Theorem IV.2.4); our analysis does not assume Re is small enough, although the iteration count of the method may deteriorate when that parameter increases.

6 A Schwarz Preconditioner with a New Coarse Space

We propose a Schwarz preconditioner for $B(\cdot, \cdot)$, by viewing $B(\cdot, \cdot)$ restricted to $P_{0,\nabla}^N(\Omega)$ as a perturbation of the symmetric bilinear form $a(\cdot, \cdot)$. We assume that the coarse triangulation $\tau_H = \cup_{i=1}^M \Omega_i$ is a shape regular triangulation, not necessarily quasi-uniform, and set $H = \max_i H_i$, where H_i is the diameter of Ω_i.

We start the definition of our coarse space by first defining an extension operator $\tilde{I}_H^h$, similar to the operator I_H^h, defined in (6). Let $\tilde{I}_H^h : V_{\mathbf{n},\nabla_H}^H(\Omega) \longrightarrow P_{0,\nabla}^N(\Omega)$, and let $\tilde{\mathbf{u}}_H = \tilde{I}_H^h(\mathbf{u}_H)$ for $\mathbf{u}_H \in V_{\mathbf{n},\nabla_H}^H(\Omega)$. The restriction of $\tilde{\mathbf{u}}_H$ to a subregion Ω_i is the solution of the following non-homogeneous Stokes problem:
Find $\tilde{\mathbf{u}}_H \in P_\nabla^N(\Omega_i)$, with $\tilde{\mathbf{u}}_H = \mathbf{u}_H$ on $\partial\Omega_i$, and $\tilde{p}_H \in P^{N-2}(\Omega_i) \cap L_0^2(\Omega_i)$ such that

$$\left\{ \begin{array}{l} a(\tilde{\mathbf{u}}_H, \mathbf{v}_N) + b(\mathbf{v}_N, \tilde{p}_H) = a(\mathbf{u}_H, \mathbf{v}_N) \quad \forall \mathbf{v}_N \in P_0^N(\Omega_i), \\ \qquad b(\tilde{\mathbf{u}}_H, q_{N-2}) = 0 \quad \forall q_{N-2} \in P^{N-2}(\Omega_i) \cap L_0^2(\Omega_i). \end{array} \right. \tag{11}$$

By restricting the test function $\mathbf{v}_N$ to have zero discrete divergence, i.e. $\mathbf{v}_N \in P^N_{0,\nabla}(\Omega_i)$, $\tilde{\mathbf{u}}_H$ can also be determined by:
Find $\tilde{\mathbf{u}}_H \in P^N_{\nabla}(\Omega_i)$, $\tilde{\mathbf{u}}_H = \mathbf{u}_H$ on $\partial\Omega_i$, and such that

$$a(\tilde{\mathbf{u}}_H, \mathbf{v}_N) = a(\mathbf{u}_H, \mathbf{v}_N) \quad \forall \mathbf{v}_N \in P^N_{0,\nabla}(\Omega_i). \tag{12}$$

The new coarse space is defined by:

$$\tilde{V}^H_{\mathbf{n},\nabla_H}(\Omega) = \tilde{I}^h_H(V^H_{\mathbf{n},\nabla_H}(\Omega)).$$

An easy argument using Green's formula shows that $\tilde{V}^H_{\mathbf{n},\nabla_H}(\Omega) \subset P^N_{0,\nabla}(\Omega)$; it is also easy to see that $\tilde{\mathbf{u}}_H$ is the function of $P^N_{0,\nabla}(\Omega)$ which coincides with $\mathbf{u}_H$ on Γ, and which is the best approximation of $\mathbf{u}_H$ in the $a(\cdot,\cdot)$-semi-norm (and in the H^1-semi-norm, since they differ only by a fixed factor ν).

The operator $Q_H : P^N_{0,\nabla}(\Omega) \rightarrow \tilde{V}^H_{\mathbf{n},\nabla_H}(\Omega)$ is defined by

$$B(Q_H\mathbf{u}, \mathbf{v}_H) = B(\mathbf{u}, \mathbf{v}_H) \quad \forall \mathbf{v}_H \in \tilde{V}^H_{\mathbf{n},\nabla_H}(\Omega). \tag{13}$$

We remark that although $B(\cdot,\cdot)$ is not necessarily positive definite, (13) is guaranteed to have solutions for sufficiently small values of H; see property **P3** below.

Let V_s, $s \geq 1$ be the local spaces used to define the operator $T_{\mathbf{n}}$; see (7). In three dimensions, there is one local space associated with the interior of each Ω_i, one space related to each face, and one for each edge. For $s \geq 1$, the operator $P_s : P^N_{0,\nabla}(\Omega) \rightarrow V_s$ is defined by

$$a(P_s\mathbf{u}, \mathbf{v}_s) = B(\mathbf{u}, v_s) \quad \forall \mathbf{v}_s \in V_s. \tag{14}$$

Theorem 6.1 *There exists a positive constant H_0, depending only on the domain Ω and on the solution $\tilde{\mathbf{u}}_N$, and positive constants $c(H_0)$, and $C(H_0)$ such that the operator*

$$Q_a = Q_H + \sum_{s\geq 1} P_s$$

satisfies, $\forall \mathbf{u} \in P^N_{0,\nabla}(\Omega)$, and for $H \leq H_0$,

$$a(Q_a\mathbf{u}, Q_a\mathbf{u}) \leq C(H_0)a(\mathbf{u},\mathbf{u}),$$

and

$$c(H_0)C_0^{-2}a(\mathbf{u},\mathbf{u}) \leq a(Q_a\mathbf{u},\mathbf{u}).$$

The proof of this result is given in [Cas96].

This estimate immediately implies an upper bound on the iteration count of the GMRES method applied to the preconditioned system

$$Q_{\mathrm{a}}\, \underline{\mathbf{u}}_N = \underline{b},$$

where $\underline{b}$ is chosen so that $\underline{\mathbf{u}}_N$ is the vector of nodal values of $\mathbf{u}_N$. This result is an extension to the Navier-Stokes equation of the Schwarz method for scalar second-order non-symmetric problems studied in [CW92].

Acknowledgement

This work has been supported in part by a Brazilian graduate student fellowship from CNPq, in part by the National Science Foundation under Grant NSF-CCR-9503408, and in part by the U. S. Department of Energy under contract DE-FG02-92ER25127.

REFERENCES

[BP89] Bramble J. H. and Pasciak J. E. (1989) A domain decomposition technique for Stokes problems. *Applied Numerical Mathematics* 6: 251–261.

[Bre94] Brenner S. C. (1994) A two-level additive Schwarz preconditioner for the stationary Stokes equations. Technical report, University of South Carolina.

[Cai95] Cai X.-C. (1995) The use of pointwise interpolation in domain decomposition methods with non-nested meshes. *SIAM J. Sci Comput.* 16(1): 250–256.

[Cas96] Casarin M. A. (1996) *Schwarz Preconditioners for Spectral and Mortar Finite Element Methods with Applications to Incompressible Fluids.* PhD thesis, Courant Institute of Mathematical Sciences. Tech. Rep. 717, Department of Computer Science, Courant Institute.

[CW92] Cai X.-C. and Widlund O. (1992) Domain decomposition algorithms for indefinite elliptic problems. *SIAM J. Sci. Statist. Comput.* 13(1): 243–258.

[Elm94] Elman H. (1994) Multigrid and Krylov subspace methods for the discrete Stokes equations. Technical report, University of Maryland. Technical Report UMIA CS-TR-94-76.

[FR94] Fischer P. F. and Rønquist E. (1994) Spectral element methods for large scale parallel Navier-Stokes calculations. *Comput. Methods Appl. Mech. Engrg* 116: 69–76. Proceedings of ICOSAHOM 92, a conference held in Montpellier, France, June 22-26, 1992.

[GPAR86] Girault V. and Pierre-Arnaud Raviart (1986) *Finite Element Methods for Navier-Stokes Equations.* Springer-Verlag, New York.

[Kla94] Klawonn A. (1994) An optimal preconditioner for a class of saddle point problems with a penalty term. Technical Report 676, Courant Institute of Mathematical Sciences, New York University.

[MPR92] Maday Y., Patera A. T., and Rønquist E. M. (1992) The $P_N \times P_{N-2}$ method for the approximation of the Stokes problem. Technical Report 92009, Université Pierre et Marie Curie, Paris, France. To appear in Numer. Math.

[Røn95] Rønquist E. M. (1995) A domain decomposition solver for the steady Navier-Stokes equations. In *Proceedings of the 1995 ICOSAHOM conference on higher order methods.* To appear.

[TP95] Tallec P. L. and Patra A. (1995) Nonoverlapping domain decomposition methods for Stokes problems with discontinuous pressure fields. Personal Communication.

9

Incomplete Domain Decomposition LU Factorizations

J. C. Díaz, M. Komara, and J. Hensley

1 Introduction

The incomplete domain decomposition LU factorizations for the solution of systems of linear equations arising from the discretization of two-dimensional non selfadjoint PDEs are introduced. The construction of the factorizations is presented for positive definite M-matrices. The theoretical discussion is for two subdomains. Multidomain numerical illustrations are also included.

Consider a decomposition of the computational domain Ω into two overlapping subregions, arbitrarily ordered Ω_1 and Ω_2. The original method due to Schwarz [Sch70] consisted of alternating the solution on each subdomain until convergence was achieved. Domain decomposition methods have evolved this idea to the construction of preconditionings. Consider LU factorizations on each subdomain. Using these factorizations, a symmetrized domain decomposition preconditioning solves in domain Ω_1, then solves in domain Ω_2, and finally corrects in domain Ω_1, see [BW86]. The cost per iteration is the cost of 3 LU solves.

The method proposed here has the feel of an LU factorization: forward elimination followed by back substitution. First using the subdomains LU factorizations, forward eliminate in domain Ω_1, carry that information to domain Ω_2 forward eliminate there. Then, the back substitution is completed in the reverse order: first, in domain Ω_2 and then in the original domain Ω_1. The cost per iteration is equivalent to the cost of 2 LU solves. Thus, the cost per iteration of the incomplete domain decomposition LU factorizations is approximately 2/3 of the cost of traditional domain decomposition factorizations.

Just as the original idea of Schwarz and the multiplicative domain decomposition methods have the feel of a Gauss-Seidel iteration on the subdomains, the factorizations proposed here have the feel of a block symmetric Gauss-Seidel. This should make the factorizations proposed here somewhat more robust than traditional domain decomposition factorizations. This is born out in the application to time dependent problems where the step size is adaptively changed for the accuracy of the solution, [Kom96]. Incomplete domain decomposition LU factorizations are able to solve the

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

linear systems for larger time steps.

The combination of less cost per iteration and robustness makes this factorization an attractive preconditioner. The incomplete domain decomposition LU factorizations can be extended to multiple subdomains, [DK97]. Furthermore, the multiple subdomains factorization is parallelizable through the use of coloring.

In Section 2, the domain Ω is decomposed into overlapping subdomains and the incomplete domain decomposition factorization is derived. A brief analysis of the factorization is presented in Section 3. The factorization is related to a regular splitting of an expanded matrix, whose dimensions exceed the original matrix according to the amount of overlap. Section 4 reports the results of some numerical experiments illustrating the potential of the factorization.

2 Incomplete Domain Decomposition *LU* Factorizations

The presentation centers in the solution of the linear system

$$Ax = b \tag{1}$$

arising from the finite difference discretization of two-dimensional PDEs on a rectangular domain Ω. A is an $n \times n$ nonsingular matrix, b is a given n-dimensional vector, and x is the n-dimensional unknown vector. The matrix A is assumed to be a positive definite M-matrix. The linear system will be solved using preconditioned conjugate gradient type methods [SSF95, VdV92].

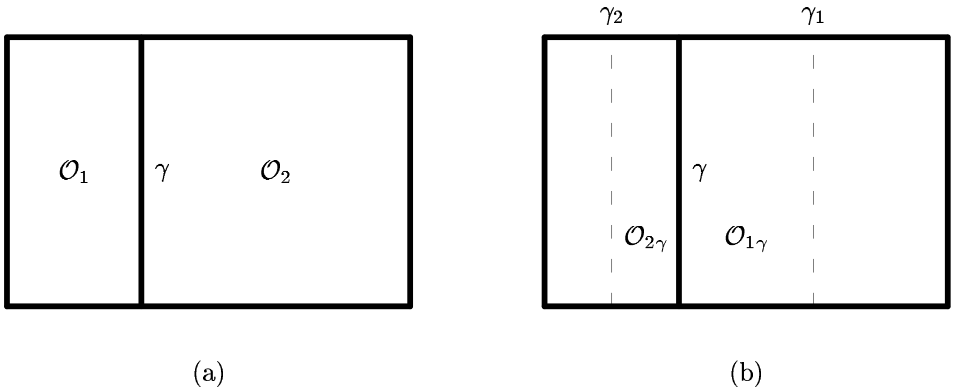
The construction of incomplete domain decomposition LU factorizations is presented for the case of two overlapping subdomains. The extension to several subdomains will be presented elsewhere due to space limitations.

Start by first subdividing Ω into two overlapping subdomains. Then the matrices constructed from the discretization of the restriction of the PDEs on these subdomains are used to construct a matrix G that has a dimension larger than that of A. The incomplete factorizations of A are obtained from the incomplete LU factorizations of G.

Decompose Ω into two non overlapping subdomains $\mathcal{O}_1$ and $\mathcal{O}_2$, and an internal boundary γ such that $\Omega = \mathcal{O}_1 \cup \gamma \cup \mathcal{O}_2$; see Fig. 1a. The internal boundary γ is extended to create subdomains Ω_1 and Ω_2 that overlap and cover Ω. Extend γ to the right and denote the new boundary by γ_1 and the region between γ and γ_1 by $\mathcal{O}_{1\gamma}$. Similarly, extend γ to the left and denote the new boundary by γ_2 and the region between γ and γ_2 is denoted by $\mathcal{O}_{2\gamma}$. The two overlapping regions are defined by $\Omega_1 = \mathcal{O}_1 \cup \mathcal{O}_{1\gamma} \cup \gamma_1$ and $\Omega_2 = \mathcal{O}_2 \cup \mathcal{O}_{2\gamma} \cup \gamma_2$, and the overlap between them is $\Omega_o = \gamma_1 \cup \mathcal{O}_{1\gamma} \cup \gamma \cup \mathcal{O}_{2\gamma} \cup \gamma_2$; see Fig. 1b. Also, let $\Omega_u = \Omega_2 \setminus \Omega_1$ and $\Omega_r = \Omega_2 \setminus \Omega_u$. Then Ω_1 and Ω_u are disjoint and cover Ω, and Ω_r and Ω_u are disjoint subdomains covering Ω_2. It can be seen that $\Omega_o = \Omega_r$.

Now let ω be the set of grid points introduced in Ω after discretizing Ω with mesh size h. Define by $\omega_1 = \omega \cap \Omega_1$ the set of grid points in Ω_1, $\omega_2 = \omega \cap \Omega_2$ the set of grid points in Ω_2, $\omega_u = \omega \cap \Omega_u$ the set of grid points in Ω_u, and $\omega_r = \omega \cap \Omega_r$ the set of grid points in Ω_r. Note that $\omega = \omega_1 \cup \omega_u$ since Ω_1 and Ω_u are disjoint subdomains covering Ω. Note also that $\omega_2 = \omega_u \cup \omega_r$ since Ω_u and Ω_r are disjoint subdomains covering Ω_2. Denote by n_1, n_2, n_u, and n_r the number of grid points in ω_1, ω_2, ω_u, and ω_r. The

Figure 1 (a) Nonoverlapping subdomains and (b) Overlapping subdomains



order of the matrix A of Equation (1), n, is equal to the number of grid points in ω, i.e. $n = n_1 + n_u$.

Let G_{11}, G_{22}, A^u_{22}, and A^r_{22} be the matrices arising from the discretization of the restriction of the PDEs on ω_1, ω_2, ω_u, and ω_r, respectively. The matrix G_{22} can be represented in 2×2 block form as

$$G_{22} = \begin{array}{cc} \begin{bmatrix} A^r_{22} & A^{ru}_{22} \\ A^{ur}_{22} & A^u_{22} \end{bmatrix} & \begin{array}{c} \omega_r \\ \omega_u \end{array} \end{array}$$

since ω_r and ω_u are disjoint subsets of ω_2 such that $\omega_2 = \omega_r \cup \omega_u$. Similarly, ω_1 and ω_u are disjoint subsets of ω such that $\omega = \omega_1 \cup \omega_u$, and hence, the matrix A can also be represented in 2×2 block form as

$$A = \begin{array}{cc} \begin{bmatrix} A_{11} & A^u_{12} \\ A^u_{21} & A^u_{22} \end{bmatrix} & \begin{array}{c} \omega_1 \\ \omega_u \end{array} \end{array}, \tag{2}$$

where

$$A_{11} = G_{11}, \quad A^u_{12} = \begin{array}{cc} \begin{bmatrix} 0 \\ A^{ru}_{22} \end{bmatrix} & \begin{array}{c} \omega_1 \setminus \omega_r \\ \omega_r \end{array} \end{array}, \quad \text{and} \quad A^u_{21} = \begin{array}{cc} \omega_1 \setminus \omega_r & \omega_r \\ [\, 0 & A^{ur}_{22} \,] \end{array}.$$

Let I_k be the identity matrix of order k and $m = n_1 + n_2$. Consider P_1, P_2 and P, respectively $n_1 \times n_1$, $n_2 \times n_u$ and $m \times n$ matrices given by

$$P_1 = I_{n_1}, \quad P_2 = \begin{array}{cc} \begin{bmatrix} 0 \\ I_{n_u} \end{bmatrix} & \begin{array}{c} \omega_r \\ \omega_u \end{array} \end{array}, \quad \text{and} \quad P = \begin{array}{cc} \begin{bmatrix} P_1 & 0 \\ 0 & P_2 \end{bmatrix} & \begin{array}{c} \omega_1 \\ \omega_2 \end{array} \end{array}. \tag{3}$$

Let G_{12}, G_{21} and G be respectively $n_1 \times n_2$, $n_2 \times n_1$ and $m \times m$ matrices defined by

$$G_{12} = \begin{array}{cc} \omega_r & \omega_u \\ [\, 0 & A^u_{12} \,] \end{array}, \quad G_{21} = \begin{array}{cc} \begin{bmatrix} 0 \\ A^u_{21} \end{bmatrix} & \begin{array}{c} \omega_r \\ \omega_u \end{array} \end{array}, \text{ and } G = \begin{array}{cc} \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} & \begin{array}{c} \omega_1 \\ \omega_2 \end{array} \end{array}. \tag{4}$$

Then, the identities hold

$$A^u_{12} = P_1^T G_{12} P_2, \quad A^u_{21} = P_2^T G_{21} P_1, \quad \text{and} \quad A^u_{22} = P_2^T G_{22} P_2.$$

Furthermore, it can readily be checked that the equality $A = P^T G P$ holds.

A is an M-matrix and so are G_{11} and G_{22} which are principal submatrices of A. It follows that the matrix

$$\widetilde{G}_{22} = \begin{bmatrix} A^r_{22} & 0 \\ A^{ur}_{22} & A^u_{22} \end{bmatrix}$$

obtained by setting some of the off-diagonal entries of G_{22} to zero is also an M-matrix. Therefore, there exist traditional splittings [BP94] of G_{11} and $\widetilde{G}_{22}$ such that

$$G_{11} = Q_1 - E_1 \quad \text{and} \quad \widetilde{G}_{22} = Q_2 - E_2,$$

where Q_1^{-1}, Q_2^{-1}, E_1 and E_2 are nonnegative matrices, i.e. the entries of Q_1^{-1}, Q_2^{-1}, E_1 and E_2 are all nonnegative. The matrices Q_1 and Q_2 are derived from the (block) ILU factorizations of G_{11} and $\widetilde{G}_{22}$ and have the form

$$Q_1 = (L_1 + B_1)B_1^{-1}(B_1 + U_1) \quad \text{and} \quad Q_2 = (L_2 + B_2)B_2^{-1}(B_2 + U_2),$$

where L_1 and L_2 are the strictly lower parts of G_{11} and $\widetilde{G}_{22}$; and U_1 and U_2 are the strictly upper parts of G_{11} and $\widetilde{G}_{22}$. The matrices B_1 and B_2 are M-matrices constructed during the factorization process.

Now let B, L, U, Q and $\widetilde{G}$ be matrices of order m defined by

$$B = \begin{array}{cc} \begin{bmatrix} B_1 & 0 \\ 0 & B_2 \end{bmatrix} & \begin{array}{c} \omega_1 \\ \omega_2 \end{array} \end{array}, \quad L = \begin{array}{cc} \begin{bmatrix} L_1 & 0 \\ G_{21} & L_2 \end{bmatrix} & \begin{array}{c} \omega_1 \\ \omega_2 \end{array} \end{array}, \quad \text{and} \quad U = \begin{array}{cc} \begin{bmatrix} U_1 & G_{12} \\ 0 & U_2 \end{bmatrix} & \begin{array}{c} \omega_1 \\ \omega_2 \end{array} \end{array}$$

$$Q = (L + B)B^{-1}(B + U) \quad \text{and} \quad \widetilde{G} = \begin{array}{cc} \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & \widetilde{G}_{22} \end{bmatrix} & \begin{array}{c} \omega_1 \\ \omega_2 \end{array} \end{array}. \tag{5}$$

The incomplete domain decomposition preconditioner of A is defined by

$$Q_{IDD} = (P^T Q^{-1} P)^{-1} \tag{6}$$

where Q and P are given in Equation (5) and (3), respectively. Note that the preconditioner has the feel of an LU factorization. Computing the action of Q_{IDD}^{-1} on a vector requires a forward elimination followed by a back substitution. From Equations (2), (3) and (4) it follows that the matrix PAP^T can be written as

$$\begin{aligned} PAP^T &= \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & \begin{bmatrix} 0 & 0 \\ 0 & A^u_{22} \end{bmatrix} \end{bmatrix} \\ &= G - G_{ur} - G_{ru}, \end{aligned} \tag{7}$$

where

$$G_{ur} = \begin{bmatrix} 0 & 0 \\ 0 & \begin{bmatrix} A^r_{22} & 0 \\ A^{ur}_{22} & 0 \end{bmatrix} \end{bmatrix} \quad \text{and} \quad G_{ru} = \begin{bmatrix} 0 & 0 \\ 0 & \begin{bmatrix} 0 & A^{ru}_{22} \\ 0 & 0 \end{bmatrix} \end{bmatrix}.$$

Note that $G_{ur}P = 0$ and $\widetilde{G} = G - G_{ru}$.

3 Analysis

The matrix $\widetilde{G}$ was constructed, in the previous section, from principal submatrices of the matrix A which has been assumed to be a positive definite M-matrix. From these assumptions the following Lemma can be established [Kom96, DK97].

Lemma 1 *There exists a matrix E such that $\widetilde{G} = Q - E$ is a regular splitting, i.e. the entries of Q^{-1} and E are all nonnegative.*

The stability of the incomplete domain decomposition factorization is established in the following Theorem.

Theorem 1 *The preconditioned system $\mathcal{K} = Q_{IDD}^{-1} A$ is a principal submatrix of $Q^{-1}\widetilde{G}$ given by $\mathcal{K} = P^T Q^{-1} \widetilde{G} P$, and all of the eigenvalues of the preconditioned system $\mathcal{K}$ have positive real part.*

Proof: First note that $P^T P = I_n$, $G_{ur} P = 0$ and $\widetilde{G} = G - G_{ru}$. Using these and Equations (6) and (7), it follows

$$\begin{aligned}
\mathcal{K} &= Q_{IDD}^{-1} A = P^T Q^{-1} P A = P^T Q^{-1} P A P^T P \\
&= P^T Q^{-1} (P A P^T) P = P^T Q^{-1} (G - G_{ur} - G_{ru}) P = P^T Q^{-1} (G - G_{ru}) P \\
&= P^T Q^{-1} \widetilde{G} P.
\end{aligned}$$

This establishes the first part of the Theorem.

Using Lemma 1 and the above result, $\mathcal{K}$ can be rewritten as

$$\begin{aligned}
\mathcal{K} &= P^T Q^{-1} \widetilde{G} P = P^T Q^{-1} (Q - E) P = P^T (I_m - Q^{-1} E) P \\
&= I_n - P^T Q^{-1} E P.
\end{aligned}$$

Since $\widetilde{G} = Q - E$ is a regular splitting, it follows that the spectral radius $\rho(Q^{-1}E)$ of $Q^{-1}E$ is less than unity ([BP94], page 181). Also, since $P^T Q^{-1} E P$ is a principal submatrix of the nonnegative matrix $Q^{-1}E$ it follows that $\rho(P^T Q^{-1} E P) \leq \rho(Q^{-1}E)$ ([BP94], page 28). Finally, the spectral radius $\rho(I_n - \mathcal{K})$ of $I_n - \mathcal{K}$ satisfies the inequality

$$\rho(I_n - \mathcal{K}) = \rho(P^T Q^{-1} E P) \leq \rho(Q^{-1} E) < 1,$$

which shows that all the eigenvalues of $\mathcal{K}$ have positive real parts. *QED*

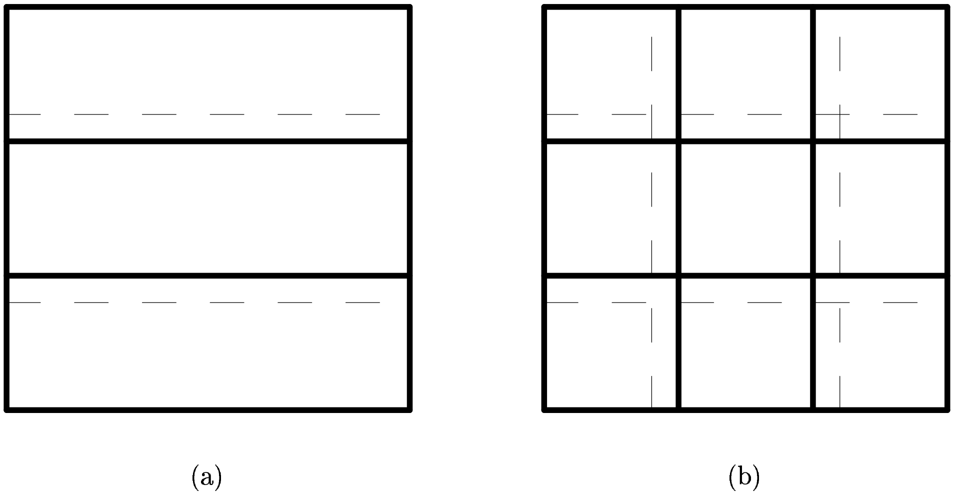
4 Numerical Experiments

The potential of the domain decomposition preconditioners is best illustrated by applying it to cases where the domain Ω has been decomposed into several subdomains. Both box and stripe decompositions of the computational domain Ω are considered; see Fig. 2.

The coefficient matrix of Equation (1) is obtained from the discretization of the PDEs on the unit square $\Omega = (0,1) \times (0,1)$. The following PDE is solved

$$-\Delta u + \gamma \left(x \frac{\partial u}{\partial x} + y \frac{\partial u}{\partial y} \right) + \beta u = f \quad \text{in} \quad \Omega$$

Figure 2 (a) Stripe decomposition (b) Box decomposition. Non overlapping and enlarged subdomains



with Dirichlet boundary conditions where $\gamma = 1000$ and $\beta = -100$. The function f is chosen such that the exact solution is $u = x(1-x)y(1-y)\exp(y)$.

The five-point finite difference scheme is used for the discretization of the PDE on a uniform grid. The first and second order derivatives are approximated using centered differences. Note that although this problem is highly non symmetric, its discretization matrix remains a positive definite M-matrix.

For $n = 32, 64, 128$, a uniform grid is introduced with spacing $h = 1/(n+1)$ in Ω. The matrix A arising from the discretization of the above PDE is a nonsingular M-matrix of order n^2 for each problem.

The linear system $Ax = b$ obtained from the discretization of the PDE is solved using preconditioned Bi-CGSTAB [VdV92] and GMRES(50) methods. The latter is the GMRES method [SSF95] that is restarted after every 50 iterations. The iterative solvers are considered to have converged when the initial residual is reduced by a factor of at least 10^{-6}, that is, the stopping criterion is $\| r_i \|_2 \leq 10^{-6} \| r_0 \|_2$, where $r_i = b - Ax_i$ is the i^{th} residual, x_i is the i^{th} approximation to the solution x, and $\| \bullet \|_2$ is the Euclidean norm. The initial guess is $x_0 = 0$ in all the test runs. The preconditioners used are the incomplete domain decomposition LU factorizations presented in this paper. To construct the preconditioner, compute the block ILU factorizations of the coefficient matrices derived from the discretization of the restriction of the PDE on the overlapping subdomains. The incomplete factorizations for these local matrices are their $INV(1)$ factorizations [CGM85, CM86, Meu89]. The ordering is the natural order. No effort is made to select a particular ordering for the grid points or for the subdomains.

The performance of the preconditioner Q_{IDD} is investigated. Throughout, the Bi-CGSTAB and GMRES(50) used in conjunction with a preconditioning matrix C will

Table 1 Number of iterations required for various grid sizes and overlaps

	Box Decompositions							Stripe Decompositions						
		$n = 32$		$n = 64$		$n = 128$			$n = 32$		$n = 64$		$n = 128$	
Ov	DM	Bi	GM	Bi	GM	Bi	GM	DM	Bi	GM	Bi	GM	Bi	GM
0h	1	4	5	6	8	11	14	1	4	5	6	8	11	14
2h	4	4	5	6	8	10	14	2	4	5	6	8	11	14
4h		4	5	6	8	10	14		4	5	6	8	11	14
6h		4	5	6	8	10	14		4	5	6	8	11	14
8h		4	5	6	8	10	14		4	5	6	8	11	14
2h	16	5	6	7	9	11	16	4	4	6	7	9	11	15
4h		5	6	7	9	11	16		4	6	7	9	11	15
6h		5	6	7	9	11	16		5	6	7	9	11	15
8h				7	9	11	16				7	9	11	15
2h	64	8	10	8	12	13	19	8	5	7	7	10	10	16
4h				8	12	13	19				7	10	11	16
6h				9	12	13	19				7	10	11	16
8h						13	19						11	16
2h	256			14	18	19	25	16			10	13	15	19
4h						19	24						16	19
6h						19	24						16	19

be denoted by Bi-CGSTAB/C and the GMRES(50)/C, respectively.

A test is carried out for obtaining the solution of the above problem using the Bi-CGSTAB/Q_{IDD} and GMRES(50)/Q_{IDD} solvers. The numerical calculations were carried out in double precision on a Sun workstation. All calculations are serial. The numerical performance of the preconditioners is considered herein. Their parallel implementations which will be presented elsewhere.

Results

The test results are gathered in Table 1. The overlap between the subdomains is labeled Ov and is the same for any two subdomains that overlap. For instance, if Ov $= n_{ov}h$, where n_{ov} is a nonnegative integer, then the overlap between any two overlapping subdomains is $n_{ov}h$. In other words, the overlap between the grids corresponding to any two overlapping subdomains is n_{ov} grid lines. The number of subdomains is reported in the column labeled DM. The number of iterations taken by Bi-CGSTAB and GMRES(50) methods are reported in columns labeled Bi and GM, respectively.

In all the test runs, the case DM $= 1$ corresponds to using the $INV(1)$ factorization of A as preconditioner; i.e. Bi-CGSTAB/$INV(1)$ and GMRES(50)/$INV(1)$ methods are used.

In all the test runs, the number of iterations seems to be independent of the size of the overlap. On the other hand, Bi-CGSTAB/Q_{IDD} and GMRES(50)/Q_{IDD}

require more iterations as the number of subdomains increases. Preconditioners based on box decompositions take more iterations than those derived from stripe decompositions. The coefficient matrices of the subdomains, however, are larger for stripe decompositions than for box decompositions. Therefore applying the preconditioners requires more computation on the subdomains in the stripe decompositions case than the box decompositions case.

Acknowledgement

Research partly supported by Oklahoma Center for Advancement of Science and Technology Grants RB9-008 (3748) and ARO-36 (3910).

REFERENCES

[BP94] Berman A. and Plemmons R. J. (1994) *Nonnegative Matrices in the Mathematical Sciences.* SIAM, Philadelphia.

[CGM85] Concus P., Golub G., and Meurant G. (1985) Block preconditioning for the conjugate gradient method. *SIAM J. Sci. Stat. Comput.* 6: 220–252.

[CM86] Concus P. and Meurant G. (1986) On computing *INV* block preconditionings for the conjugate gradient method. *BIT* 26: 493–504.

[DK97] Díaz J. C. and Komara M. (1997) The incomplete domain decomposition *lu* factorizations. *Submitted* .

[Kom96] Komara M. (1996) Incomplete multilevel *LU* factorizations. Ph.D. Dissertation, Center for Parallel and Scientific Computing, The University of Tulsa.

[Meu89] Meurant G. (1989) Incomplete domain decomposition preconditioners for nonsymmetric problems. In Chan T. F., Glowinski R., Periaux J., and Widlund O. B. (eds) *Proceedings of the Second International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 219–225. SIAM, Philadelphia.

[SBG96] Smith B., Bjørstad P., and Gropp W. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge.

[Sch70] Schwarz H. A. (1870) Vierteljahrsschrift der naturforschenden gesellschaft. *Ges. Zurich* 15: 272–286.

[SS86] Saad Y. and Schultz M. H. (1986) GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* 10: 36–52.

[vdV92] van der Vorst H. A. (1992) Bi–CGSTAB: A fast and smoothly converging variant of Bi–CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* 13: 631–644.

10

An Additive Schwarz method for Elliptic Mortar Finite Element Problems in Three Dimensions

Maksymilian Dryja

1 Introduction

In this paper, we discuss a domain decomposition method for solving linear systems of algebraic equations arising from the discretization of elliptic problem in 3-D by the mortar element method; see [Mar90, AG93] and literature given therein. The elliptic problem is second-order with discontinuous coefficients and the Dirichlet boundary condition. Using the framework of the mortar method, the problem is approximated by the finite element method with piecewise linear functions on nonmatching meshes.

The domain decomposition method is of iterative substructuring type and is described as an additive Schwarz method (ASM) using the general framework of ASMs; see [DW95, Ben95a]. It is applied to the Schur complement of our discrete problems, i.e., interior variables of all subregions are first eliminated using a direct method.

In this paper, we consider the mortar element method in the geometrically conforming case only. The region Ω is a union of simplices Ω_i on which a coefficient ρ_i of the problem is constant. The described ASM uses a standard coarse space defined on the triangulation formed by Ω_i of diameter H, i.e., $V_0 = V^H$, a space of piecewise linear continuous functions which vanish on $\partial\Omega$.

The algorithm described is almost optimal, i.e., the number of iterations required to decrease the energy norm of the error in a conjugate gradient method is proportional to $(1 + log\frac{H}{h})^g$, where $g = 1$ or $g = \frac{3}{2}$ with the constant independent of the coarse and fine meshes (H and h) and the coefficient ρ_i. This result is proved assuming a special distribution of the ρ_i on Ω_i, called quasi-monotone (introduced in [DSW96]) and weak quasi-monotone (introduced herein). This is the main result of the paper. There are indications that this result is sharp in 3-D with respect to the distribution of ρ_i, see [Osw95] and [Xu91]. In the case of arbitrary distribution of ρ_i, the number of iterations can be bounded by $(H/h)^{\frac{1}{2}}$.

The results of this paper are generalizations of results obtained in [Ben95b] for 2-D. The idea of using a standard coarse space is taken from [Glo84], where the 2-D case

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

with regular coefficients is considered. The mortar element method in the geometrically nonconforming case for problems with discontinuous coefficients is not discussed here. The reason is that it is not clear how to design and analyze ASMs for either the standard coarse space or for others; see, for example, the new coarse space used in [Ben95b] in the 2-D case.

The outline of the paper is as follows. In Section 1.2, the discrete problem obtained from the mortar element method is described. In Section 1.3, an iterative substructuring method is described in terms of an ASM for the Schur complement system. In this section, Theorem 1.3.1 is formulated as the main result of the paper. A proof of this theorem is given in Section 1.5. In Section 1.4, technical tools are given which are needed for the proof.

Some of the results of this paper have been obtained in joint work with Olof Widlund.

2 Mortar Discrete Problems

We solve the following differential problem: Find $u^* \in H_0^1(\Omega)$ such that

$$a(u^*, v) = f(v), \quad v \in H_0^1(\Omega), \tag{1}$$

where

$$a(u,v) = \sum_i \rho_i (\nabla u, \nabla v)_{L^2(\Omega_i)}, \quad f(v) = (f,v)_{L^2(\Omega)},$$

$\bar{\Omega} = \cup_{i=1}^N \bar{\Omega}_i$, and ρ_i is a positive constant in Ω_i.

Let Ω be a polygonal region in 3-D and Ω_i be tetrahedral elements. They form a coarse triangulation with a parameter H. In each Ω_i, a triangulation is introduced with tetrahedral elements $e_j^{(i)}$ and a parameter h_i. The resulting triangulation of Ω is nonmatching. We assume that the coarse triangulation and the h_i triangulation in each Ω_i are quasi-uniform, see [GPP94]. Let $X_i(\Omega_i)$ be the finite element space of piecewise linear continuous functions defined on the triangulation of Ω_i and vanishing on $\partial\Omega_i \cap \partial\Omega$. Let

$$X^h(\Omega) = X_1(\Omega_1) \times \cdots \times X_N(\Omega_N).$$

To define the mortar finite element method, we introduce some notation and spaces. Let

$$\Gamma = (\cup_i \partial\Omega_i) \backslash \partial\Omega$$

and let F_{ij}, E_{ij} denote the faces, edges of Ω_i. The union of $\bar{E}_{ij}$ forms the wire basket W_i of Ω_i. We now select open faces γ_m of Γ, called mortars (masters) such that

$$\Gamma = \cup_m \bar{\gamma}_m \ \text{ and } \ \gamma_m \cap \gamma_n = \emptyset \ \text{ if } \ m \neq n.$$

By $\gamma_{m(i)}$ we denote a face of Ω_i. Let $\gamma_{m(i)}$ be a face common to Ω_i and Ω_j. As a face of Ω_j it is denoted by $\delta_{m(j)}$ and it is called nonmortar (slave). The rule for selecting $\gamma_{m(i)} = F_{ij}$, a common face to Ω_i and Ω_j, as mortar is that $\rho_i \geq \rho_j$. Let $W^{h_i}(F_{ij})$ be the restriction of $X_i(\Omega_i)$ to F_{ij}. Note that on $F_{ij} = \gamma_{m(i)} = \delta_{m(j)}$, the common face to Ω_i and Ω_j, we have two triangulation, denoted in terms of h_i and h_j and two different spaces $W^{h_i}(\gamma_{m(i)})$ and $W^{h_j}(\delta_{m(j)})$.

Let $M^{h_j}(\delta_{m(j)})$ denote a subspace of $W^{h_j}(\delta_{m(j)})$ defined as follows: Let $v \in W^{h_j}(\delta_{m(j)})$. A function $\tilde{v} \in M^{h_j}(\delta_{m(j)})$ has the same values as v at the interior nodal points of $\delta_{m(j)}$. The value of $\tilde{v}$ at a nodal point $x_k \in \partial\delta_{m(j)}$, the boundary of $\delta_{m(j)}$, is equal to

$$\tilde{v}(x_k) = \sum_{i=1}^{n_k} \alpha_i v(x_{i(k)}) \quad \sum_{i=1}^{n_k} \alpha_i = 1,$$

where $\alpha_i \geq 0$ and the sum is taken over interior nodal points $x_{i(k)}$ of $\delta_{m(j)}$ such that an interval $(x_k, x_{i(k)})$ is a side of the triangulation and its number is equal to n_k, for details see [AG93].

We say that $u_{i(m)}$ and $u_{j(m)}$, the restrictions of $u_i \in X_i(\Omega_i)$ and $u_j \in X_j(\Omega_j)$ to δ_m, a common face to Ω_i and Ω_j, satisfy the mortar condition if

$$\int_{\delta_m} (u_{i(m)} - u_{j(m)})\Psi ds = 0, \quad \Psi \in M^{h_j}(\delta_m). \tag{2}$$

This condition can be rewritten as follows. Let $\Pi_m(u_{i(m)}, v_{j(m)})$ denote a projection from $L^2(\delta_m)$ on $W^{h_j}(\delta_m)$ defined by

$$\int_{\delta_m} \Pi_m(u_{i(m)}, v_{j(m)})\Psi ds = \int_{\delta_m} u_{i(m)}\Psi ds, \quad \Psi \in M^{h_j}(\delta_m) \tag{3}$$

and

$$\Pi_m(u_{i(m)}, v_{j(m)})_{|\partial\delta_m} = v_{j(m)}. \tag{4}$$

Thus $u_{j(m)} = \Pi_m(u_{i(m)}, v_{j(m)})$ if $v_{j(m)} = u_{j(m)}$ on $\partial\delta_m$.

By V^h we denote a space of $v \in X^h$ which satisfies the mortar condition for each $\delta_m \subset \Gamma$. The discrete problem for (1) in V^h is defined as follows: Find $u_h^* \in V^h$ such that

$$b(u_h^*, v_h) = f(v_h), \quad v_h \in V^h, \tag{5}$$

where

$$b(u_h, v_h) = \sum_{i=1}^{N} a_i(u_{ih}, v_{ih}) = \sum_{i=1}^{N} \rho_i(\nabla u_{ih}, \nabla v_{ih})_{L^2(\Omega_i)}$$

and $v_h = \{v_{ih}\}_{i=1}^N \in V^h$. It is known that V^h is a Hilbert space with an inner product defined by $b(u, v)$. This problem has an unique solution and an estimate of the error is known, see [AG93].

3 Additive Schwarz Method

In this section, we describe an additive Schwarz method for (5). It will be given for the Schur complement system. For that we first eliminate all interior unknowns of Ω_i using for $u_i \in X_i(\Omega_i)$ (here and below we drop the index h for functions)

$$u_i = Pu_i + Hu_i, \tag{6}$$

where Hu_i is discrete harmonic in Ω_i in the sense of $(\nabla u, \nabla v)_{L^2(\Omega_i)}$ with $Hu_i = u_i$ on $\partial\Omega_i$. Using this, we get

$$s(u^*, v) = f(v), \quad v \in V^h(\Omega), \tag{7}$$

where here and below V^h denotes a space of discrete harmonic functions in each Ω_i and

$$s(u,v) = b(u,v), \quad u, v \in V^h(\Omega).$$

An additive Schwarz method (ASM) for (7) is designed and analyzed using the general ASM framework, see [Ben95a]. Using this framework, the method is designed in terms of a decomposition of V^h, certain bilinear forms given on these subspaces, and the projections onto these subspaces in the sense of these bilinear forms.

The decomposition of V^h is taken as

$$V^h(\Omega) = V_0(\Omega) + \sum_{\gamma_m \subset \Gamma} V_m^{(F)}(\Omega) + \sum_{i=1}^{N} \sum_{x_k \in W_{ih}} V_k^{(W_i)}(\Omega). \tag{8}$$

Here $V_0 = V^H$ is a space of piecewise linear continuous functions, on the coarse triangulation, which vanish on $\partial\Omega$. The space $V_m^{(F)}(\Omega)$ is a subspace of V^h associated with the master face γ_m. It is the restriction of V^h to γ_m and δ_m ($\gamma_m = \delta_m$), and the zero on $\partial\gamma_m$ and $\partial\delta_m$, the remaining master and slave faces, and on $\partial\Omega$. W_{ih} is the set of nodal points of W_i. $V_k^{(W_i)}$ is an one-dimensional space associated with $x_k \in W_{ih}$ and spanned by Φ_k. The function Φ_k is discrete harmonic with data on the boundary of the substructures defined as follows: Let x_k be a nodal point of $\partial\gamma_{m(i)}$, the boundary of the mortar face $\gamma_{m(i)}$ of Ω_i. We set $\Phi = \varphi_k(x)$ on $\gamma_{m(i)}$, where $\varphi_k(x)$ is a nodal basis function associated with x_k. Let $\delta_{m(j)} = \gamma_{m(i)} = F_{ij}$ be the face common to Ω_i and Ω_j. Φ_k is equal to $\Pi_m(\varphi_k, 0)$ on $\delta_{m(j)}$; see (3) and (4). Φ_k is defined on the remaining mortar faces of Ω_i in the same way if x_k is a nodal point of their boundaries. Φ_k is zero on the remaining mortar and nonmortar faces of Γ. Let x_k be a nodal point of $\partial\delta_{m(i)}$, the boundary of a nonmortar face of Ω_i. $\Phi_k(x)$ is equal to $\Pi_m(0, \varphi_k)$ on $\delta_{m(i)}$. This means that $\Phi_k = 0$ on the mortar face $\gamma_{m(j)} = \delta_{m(i)}$. Φ_k is defined on the remaining nonmortar faces of Ω_i in the same way if x_k is a nodal point of their boundaries. Φ_k is zero on the mortar and nonmortar faces belonging to remaining substructures. If x_k is a nodal point common to the boundaries of mortar or nonmortar faces, Φ_k is defined on these faces as above.

Let us now introduce bilinear forms defined on the introduced spaces. $b_m^{(F)}$ associated with $V_m^{(F)} \times V_m^{(F)} \to R$ is of the form

$$b_m^{(F)}(u_{m(i)}, v_{m(i)}) = (\rho_i + \rho_j)(\nabla u_{m(i)}, \nabla v_{m(i)})_{L^2(\Omega_i)}, \tag{9}$$

where $u_{m(i)}$ is the discrete harmonic function in Ω_i with data $u_{m(i)}$ on the mortar face $\gamma_{i(m)}$ of Ω_i which is common to Ω_j and zero on the remaining faces of Ω_i.

We set $b_k^{(W_i)} : V_k^{(W_i)} \times V_k^{(W_i)} \to R$ and $b_0 : V_0 \times V_0 \to R$ equal to $b(u,v)$.

Let us now introduce operators $T_m^{(F)}$, $T_k^{(W_i)}$, and T_0 by the bilinear forms $b_m^{(F)}$, $b_k^{(W_i)}$, and b_0, respectively, in the standard way. For example, $T_m^{(F)} : V^h \to V_m^{(F)}$ is the solution of

$$b_m^{(F)}(T_m^{(F)} u, v) = b(u,v), \quad v \in V_m^{(F)}. \tag{10}$$

Let

$$T = T_0 + \sum_{\gamma_m \subset \Gamma} T_m^{(F)} + \sum_{i=1}^{N} \sum_{x_k \in W_{ih}} T_k^{(W_i)}.$$

The method described is almost optimal assuming a special distribution of the coefficients ρ_i, called quasi-monotone, introduced in [DSW96]. The quasi-monotone distribution on substructures with common vertex x_k requires a monotone path from each substructure to the substructure having the largest coefficient, traversing through faces of substructures only. If the vertex $x_k \in \partial\Omega$, we additionally assume that $\partial\Omega_i \cap \partial\Omega$ contains a face of the substructure Ω_i with the largest coefficient ρ_i. This is a local condition. The distribution ρ_i in Ω is quasi-monotone if it is quasi-monotone at each vertex of the substructures; for details see [DSW96]. We also introduce the concept of a weak quasi-monotone distribution of ρ_i for which the traversing path is also allowed to go through edges. In this case, for a vertex $x_k \in \partial\Omega$, we assume that $\partial\Omega_i \cap \partial\Omega$ contains the face or the edge of Ω_i for which ρ_i is the largest in Ω_i.

There are indications that the estimates given below are sharp; see [Osw95] and [Xu91].

Theorem 3.1 *For all $u \in V^h$*

$$C_0(1 + log\frac{H}{h})^{-2}\delta^{-1}a(u,u) \leq a(Tu,u) \leq C_1 a(u,u), \tag{11}$$

where C_i are positive constants independent of H, h_i and ρ_i, $h = inf_i h_i$ and

$$\delta = \begin{cases} 1 & \text{when } \rho_i \text{ is quasi-monotone} \\ (1 + \log\frac{H}{h}) & \text{when } \rho_i \text{ is weakly quasi-monotone} \\ \frac{H}{h} & \text{when } \rho_i \text{ is not even weakly quasi-monotone} \end{cases} \tag{12}$$

4 Technical Tools

In this section, we formulate some auxiliary results that we need to prove Theorem 3.1.

Lemma 4.1 *Let $\gamma_{i(m)} = \delta_{j(m)}$ be a face common to Ω_i and Ω_j, and let $u_{i(m)}$ and $u_{j(m)}$ be the restrictions of $u_i \in X_i(\Omega_i)$ and $u_j \in X_j(\Omega_j)$ to $\gamma_{i(m)}$ and $\delta_{j(m)}$, respectively. If $u_{i(m)}$ and $u_{j(m)}$ satisfy the mortar condition (2) on $\delta_{j(m)}$ and $u_{j(m)}$ vanishes on $\partial\delta_{j(m)}$, then*

$$||u_{j(m)}||^2_{L^2(\delta_{j(m)})} \leq C||u_{i(m)}||^2_{L^2(\gamma_{i(m)})}, \tag{13}$$

where C is independent of h_i and h_j.

This lemma follows from Lemma 2.1 in [AG93].

Lemma 4.2 *Let the assumptions of Lemma 4.1 be satisfied and additionally $u_{i(m)}$ vanishes on $\partial\delta_{i(m)}$. Then,*

$$||u_{j(m)}||^2_{H^{\frac{1}{2}}_{00}(\delta_{j(m)})} \leq C||u_{i(m)}||^2_{H^{\frac{1}{2}}_{00}(\gamma_{i(m)})}, \tag{14}$$

where C is independent of h_i and h_j.

A proof of this lemma follows from Lemma 4.1 and properties of the standard L^2 projection on $W^{h_j}(\delta_{j(m)}) \cap H_0^1(\delta_{j(m)})$. In the 2-D case, Lemma 4.2 is a particular case of Lemma 1 in [Pes72]. Alternative proofs of this result, also in the 2-D case, are given in [Glo84] and [Ben95b].

Lemma 4.3 *Let Φ_k be a function defined in Section 1.3 and associated with a nodal point $x_k \in W_i$. Then*

$$b(\Phi_k, \Phi_k) \leq C\rho_i h_i, \tag{15}$$

where C is independent of h_i and ρ_i.

A proof of this lemma follows from Lemma 4.1 and the definition of Φ_k.

Let $R(x_k)$ be a union of the substructures Ω_i with a common vertex x_k.

Lemma 4.4 *For $u \in V^h$*

$$\inf_{\alpha \in R} ||u - \alpha||^2_{L^2(R(x_k))} \leq C \sum_{\Omega_i \subset R(x_k)} H^2 |u|^2_{H^1(\Omega_i)}, \tag{16}$$

where C is a positive constant independent of h_i and H.

A proof of this lemma in the 2-D case is given in [Glo84]. An alternative proof follows from

$$||u - \alpha||^2_{L^2(R(x_k))} \leq 2 \sum_{i=1}^{n_k} \left(||u - \bar{u}_i||^2_{L^2(\Omega_i)} + ||\bar{u}_i - \alpha||^2_{L^2(\Omega_i)} \right). \tag{17}$$

Here the Ω_i with a common x_k are ordered from $i = 1, \ldots, n_k$ in such a way that Ω_i and Ω_{i+1} have a common face $F_{i,i+1}$ and $\bar{u}_i$ is the average value of u_i over $F_{i,i+1}$. Using now Poincare's inequality, we get (16).

Let Q^H_ρ denote the L^2_ρ projection from V^h to $V_0 = V^H$ in the weighted inner product.

Lemma 4.5 *For $u \in V^h$*

$$b(Q^H_\rho u, Q^H_\rho u) \leq C\delta b(u, u) \tag{18}$$

and

$$||u - Q^H_\rho u||^2_{L^2_\rho(\Omega)} \leq CH^2 \delta b(u, u), \tag{19}$$

where δ is given by (12) and C is constant independent of H, h_i and ρ_i.

A proof of this lemma is a slighted modification of the proof of Lemma 9 in [DSW96].

5 Proof of Theorem 1.3.1

Using the general theorem of ASMs, we need to check three key assumptions; see [DW95] and [Ben95a].

Assumption (ii) It is shown that $\rho(\varepsilon) \leq C$ in view of Lemma 4.3.

Assumption (iii) Of course, $\omega = 1$ for $b_0(u,u),\ u \in V_0$ and $b_k^{(W_i)}(u,u),\ u \in V_k^{(W_i)}$. We now show that for $u \in V_m^{(F)}$

$$b(u,u) \leq C b_m^{(F)}(u,u). \tag{20}$$

Let $\gamma_{i(m)} = \delta_{j(m)}$ be the mortar and nonmortar sides of Ω_i and Ω_j, respectively. We have for $u \in V_m^{(F)}$

$$b(u,u) = a_i(u_i,u_i) + a_j(u_j,u_j) \leq C\big(\rho_i |u_i|^2_{H_{00}^{\frac{1}{2}}(\gamma_{i(m)})} + \rho_j |u_j|^2_{H_{00}^{\frac{1}{2}}(\delta_{j(m)})}\big).$$

Using now Lemma 4.2, we get (20) with $w = C$.

Assumption (i) We show that for $u \in V^h$, there exists a decomposition

$$u = u_0 + \sum_{\gamma_m \subset \Gamma} u_m^{(F)} + \sum_{i=1}^{N} \sum_{x_k \in W_{ih}} u_k^{(W_i)}, \tag{21}$$

where $u_0 \in V_0$, $u_m^{(F)} \in V_m^{(F)}$ and $u_k^{(W_i)} \in V_k^{(W_i)}$, such that

$$b_0(u_0,u_0) + \sum_{\gamma_m \subset \Gamma} b_m^{(F)}(u_m^{(F)}, u_m^{(F)}) + \sum_{i=1}^{N} \sum_{x_k \in W_{ih}} b_k^{(W_i)}(u_k^{(W_i)}, u_k^{(W_i)})$$
$$\leq C\delta(1 + log\frac{H}{h})^2 b(u,u). \tag{22}$$

Let $u_0 = Q_\rho^H u$, $w = u - u_0$, and w_i be the restriction of w to $\bar{\Omega}_i$. It is decomposed on $\partial\Omega_i$ as

$$w_i = \sum_{F_{ij} \subset \partial\Omega_{ih}} w_i^{(F_{ij})}(x) + w_i^{(W_i)}, \quad w_i^{(W_i)} = \sum_{x_k \in W_{ih}} w_i(x)\Phi_k, \tag{23}$$

where $w_i^{(F_{ij})}(x)$ is the restriction of $w_i - w_i^{(W_i)}$ to F_{ij}, the face of Ω_i, and zero on $\partial\Omega_i \backslash F_{ij}$.

To define $u_m^{(F)}$ let $F_{ij} = \gamma_{i(m)} = \delta_{j(m)}$ be a face common to Ω_i and Ω_j. We set

$$u_m^{(F)} = \{w_i^{(F_{ij})} \text{ on } \partial\Omega_i \text{ and } w_j^{(F_{ij})} \text{ on } \partial\Omega_j\}$$

and zero at the remaining nodal points of Γ. The function $u_k^{(W_i)}$ is defined as

$$u_k^{(W_i)} = w_i(x_k)\Phi_k(x). \tag{24}$$

It is easy to see that these functions satisfy (21).

To prove (22) note first that

$$b_0(u_0,u_0) \leq C\delta b(u,u) \tag{25}$$

by Lemma 4.5.

Let us now consider the estimate for $u_m^{(F)} \in V_M^{(F)}$ when $\gamma_{m(i)} = \delta_{m(i)} = F_{ij}$, a face common to Ω_i and Ω_j. It is known that

$$b_m^{(F)}(u_m^{(F)}, u_m^{(F)}) \leq C(\rho_i + \rho_j)||w_i^{(F_{ij})}||^2_{H_{00}^{\frac{1}{2}}(\gamma_{i(m)})}$$
$$\leq C\rho_i(1 + log\frac{H}{h_i})^2||u_i - u_0||^2_{H^1(\Omega_i)};$$

see, for example, [DW95]. We have used here also the fact that $\rho_i \geq \rho_j$. Summing with respect to γ_m and using Lemma 4.5, we get

$$\sum_{\gamma_m \subset \Gamma} b_m^{(F)}(u_m^{(F)}, u_m^{(F)}) \leq C\delta(1 + log\frac{H}{h})^2 b(u, u). \tag{26}$$

We now prove that

$$\sum_{i=1}^{N} \sum_{x_k \in W_{ih}} b_k^{(W_i)}(u_k^{(W_i)}, u_k^{(W_i)}) \leq C\delta(1 + log\frac{H}{h})b(u, u). \tag{27}$$

For that note first that, see (24),

$$b_k^{(W_i)}(u_k^{(W_i)}, u_k^{(W_i)}) \leq Cw_i^2(x_k)b(\Phi_k, \Phi_k) \leq C\rho_i h_i w_i^2(x_k)$$

in view of Lemma 4.3. Summing with respect to $x_k \in W_{ih}$, we get

$$\sum_{x_k \in W_{ih}} b_k^{(W_i)}(u_k^{(W_i)}, u_k^{(W_i)}) \leq C\rho_i||w_i||^2_{L^2(W_i)} \leq C\rho_i(1 + log\frac{H}{h_i})||w_i||^2_{H^1(\Omega_i)}.$$

Summing now with respect to i and using Lemma 4.5, we get (27).

To get (22), we add the inequalities (25), (26), and (27). The proof of Theorem 3.1 is complete.

Acknowledgement

This work was supported in part by the National Science Foundation under Grant NSF-CCR-9503408 and in part by Polish Scientific Grant 102/P03/95/09.

REFERENCES

[Bel96] Belgacem F. B. (1996) The mortar finite element method with Lagrange multipliers. *Submited to Numer. Math.* .

[BM94] Belgacem F. B. and Maday Y. (1994) Non conforming spectral method for second order elliptic problems. *East-West J. Numer. Math.* 4: 235–251.

[BMP94] Bernardi C., Maday Y., and Patera A. (1994) A new nonconforming approach to domain decomposition: The mortar element method. In Brezis H. and Lions J.-L. (eds) *College de France Seminar*. Pitman.

[Cia78] Ciarlet P. (1978) The finite element method for elliptic problems. *North-Holland, Amsterdam* .

[CW96] Casarin M. and Widlund O. (1996) A hierarchical preconditioner for the mortar finite element method. *ETNA* 4: 75–88.

[Dry96] Dryja M. (1996) Additive Schwarz methods for elliptic mortar finite element problems. In Malanowski K., Nahorski Z., and Peszynska M. (eds) *Modeling and optimization of distributed parameter systems with applications to engineering.* IFIP, Chapman and Hall, London.

[DSW94] Dryja M., Smith B., and Widlund O. (1994) Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions. *SIAM J. Numer.Anal.* 31(6): 1662–1694.

[DSW96] Dryja M., Sarkis M., and Widlund O. (1996) Multilevel Schwarz methods for elliptic problems with discontinuous coefficients in three dimensions. *Numer. Math.* 72(3): 313–348.

[DW95] Dryja M. and Widlund O. (1995) Schwarz methods of Neumann-Neumann type for three-dimensional elliptic finite element problems. *Comm. Pure Appl. Math.* 48(2): 313–348.

[Osw95] Oswald P. (1995) On the robustness of the BPX - preconditioner with respect to jumps in the coefficients. Technical Report TX 77843-3368, Department of Mathematics, Texas A & M University, College Station.

[Xu91] Xu J. (1991) Counter examples concerning a weighted L^2 projection. *Math. Comp.* 57: 563–568.

11

Overlapping Schwarz for Parabolic Problems

Martin J. Gander

1 Introduction

The basic ideas underlying waveform relaxation were first suggested in the late 19th century by Picard [Pic93] and Lindelöf [Lin94] to study initial value problems from a theoretical viewpoint. Much recent interest in waveform relaxation as a practical parallel method for the solution of stiff ordinary differential equations (ODEs) has been generated by the publication of a paper by Lelarasmee and coworkers [LRSV82] in the VLSI literature. Recent work in this field includes papers by Miekkala and Nevanlinna [MN87], Nevanlinna [Nev89, Nev90], Bellen and Zennaro [BZ93], Reichelt, White and Allen [RWA95], Jeltsch and Pohl [JP95], Burrage [Bur95] and Lumsdaine, Reichelt, Squyres and White [LRSW96].

There are two classical convergence results for waveform relaxation algorithms for ODEs: (i) for linear systems of ODEs on unbounded time intervals one can show linear convergence of the algorithm under some dissipation assumptions on the splitting; (ii) for nonlinear systems of ODEs (including linear ones) on bounded time intervals one can show superlinear convergence assuming a Lipschitz condition on the splitting function.

For classical relaxation methods (Jacobi, Gauss Seidel, SOR) the above convergence results depend on the discretization parameter if the ODE arises from a partial differential equation (PDE) which is discretized in space. The convergence rates deteriorate as one refines the mesh.

Jeltsch and Pohl propose in [JP95] a multi-splitting algorithm with overlap. They prove results (i) and (ii) for their algorithm, but the convergence rates are mesh-dependent. However they show numerically that increasing the overlap accelerates the convergence of the waveform relaxation algorithm. We quantify their numerical results by formulating the waveform relaxation algorithm at the space-time continuous level using overlapping domain decomposition; this approach was motivated by the work of Bjørhus [Bjø95]. We show linear convergence of this algorithm on unbounded time intervals at a rate depending on the size of the overlap. This is an extension of the first classical convergence result (i) for waveform relaxation from ODEs to PDEs.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Discretizing the algorithm, the size of the physical overlap corresponds to the overlap of the multi-splitting algorithm analyzed by Jeltsch and Pohl. We show furthermore that the convergence rate is robust with respect to mesh refinement, provided the physical overlap is held constant during the refinement process. The details of the analysis can be found in [GS97].

Independently Giladi and Keller [GK97] studied superlinear convergence of domain decomposition algorithms for the convection-diffusion equation on bounded time intervals, hence generalizing the second classical waveform relaxation result (ii) from ODEs to PDEs.

2 Continuous Case

Consider the one-dimensional inhomogeneous heat equation on the interval $[0, L]$,

$$\begin{array}{rcll} \frac{\partial u}{\partial t} & = & \frac{\partial^2 u}{\partial x^2} + f(x,t) & 0 < x < L, \ t > 0 \\ u(0,t) & = & g_1(t) & t > 0 \\ u(L,t) & = & g_2(t) & t > 0 \\ u(x,0) & = & u_0(x) & 0 < x < L, \end{array} \tag{1}$$

where we assume enough smoothness on the data such that (1) has a unique bounded solution [Can84]. Given any function $f(t) : \mathbb{R}^+ \longrightarrow \mathbb{R}$ we define

$$||f(\cdot)||_\infty := \sup_{t>0} |f(t)|.$$

We decompose the domain $\Omega = [0, L] \times [0, \infty)$ into two overlapping subdomains $\Omega_1 = [0, \beta L] \times [0, \infty)$ and $\Omega_2 = [\alpha L, L] \times [0, \infty)$, where $0 < \alpha < \beta < 1$. The solution $u(x, t)$ of (1) can now be obtained by composing the solutions $v(x, t)$ on Ω_1 and $w(x, t)$ on Ω_2, which satisfy the same inhomogeneous heat equation on the subdomains with the new interior boundary conditions $v(\beta L, t) = w(\beta L, t)$ and $w(\alpha L, t) = v(\alpha L, t)$, respectively. Note that $v(x, t) \equiv w(x, t)$ in the overlap. The system, which is coupled through the boundary, can be solved using an alternating Schwarz iteration, where the new function $v^{k+1}(x, t)$ on Ω_1 is obtained using the previous iterate $w^k(x, t)$ at the interior boundary and similarly on Ω_2. Let $d^k(x, t) := v^k(x, t) - v(x, t)$ and $e^k(x, t) := w^k(x, t) - w(x, t)$ and consider the error equations

$$\begin{array}{rcll} \frac{\partial d^{k+1}}{\partial t} & = & \frac{\partial^2 d^{k+1}}{\partial x^2} & 0 < x < \beta L, \ t > 0 \\ d^{k+1}(0,t) & = & 0 & t > 0 \\ d^{k+1}(\beta L,t) & = & e^k(\beta L,t) & t > 0 \\ d^{k+1}(x,0) & = & 0 & 0 < x < \beta L \end{array} \tag{2}$$

and

$$\begin{array}{rcll} \frac{\partial e^{k+1}}{\partial t} & = & \frac{\partial^2 e^{k+1}}{\partial x^2} & \alpha L < x < L, \ t > 0 \\ e^{k+1}(\alpha L,t) & = & d^k(\alpha L,t) & t > 0 \\ e^{k+1}(L,t) & = & 0 & t > 0 \\ e^{k+1}(x,0) & = & 0 & \alpha L < x < L. \end{array} \tag{3}$$

Given any function $g(x,t) : [a,b] \times \mathbb{R}^+ \longrightarrow \mathbb{R}$ we define

$$||g(\cdot,\cdot)||_{\infty,\infty} := \sup_{a<x<b,t>0} |g(x,t)|.$$

Theorem 2.1 *The Schwarz iteration for the heat equation with two subdomains converges at a rate depending on the size of the overlap. The error on the two subdomains decays at the rate*

$$\begin{aligned} ||d^{2k+1}(\cdot,\cdot)||_{\infty,\infty} &\leq \left(\frac{\alpha(1-\beta)}{\beta(1-\alpha)}\right)^k ||e^0(\beta L,\cdot)||_\infty \qquad (4)\\ ||e^{2k+1}(\cdot,\cdot)||_{\infty,\infty} &\leq \left(\frac{\alpha(1-\beta)}{\beta(1-\alpha)}\right)^k ||d^0(\alpha L,\cdot)||_\infty. \qquad (5)\end{aligned}$$

Proof The proof is obtained using the maximum principle of the heat equation and can be found in [GS97]. ■

3 Semi-Discrete Case

Consider the heat equation continuous in time, but discretized in space using a centered second order finite difference scheme on a grid with n grid points and $\Delta x = \frac{L}{n+1}$. This gives

$$\begin{aligned} \frac{\partial \boldsymbol{u}}{\partial t} &= A_{(n)}\boldsymbol{u} + \boldsymbol{f}(t) \qquad t>0\\ \boldsymbol{u}(0) &= \boldsymbol{u}_0, \end{aligned} \qquad (6)$$

where the $n \times n$ matrix $A_{(n)}$ is given by

$$A_{(n)} = \frac{1}{(\Delta x)^2}\begin{bmatrix} -2 & 1 & & 0\\ 1 & -2 & \ddots & \\ & \ddots & \ddots & 1\\ 0 & & 1 & -2 \end{bmatrix} \qquad (7)$$

and $\boldsymbol{f}(t) = (f(\Delta x,t) + \frac{g_1(t)}{(\Delta x)^2}, f(2\Delta x,t), \dots, f((n-1)\Delta x,t), f(n\Delta x,t) + \frac{g_2(t)}{(\Delta x)^2})^T$, $\boldsymbol{u}_0 = (u_0(\Delta x), \dots, u_0(n\Delta x))^T$.

We decompose the domain into two overlapping subdomains Ω_1 and Ω_2. We assume for simplicity that αL falls on the grid point $i = a$ and βL on the grid point $i = b$. We therefore have $a\Delta x = \alpha L$ and $b\Delta x = \beta L$. As in the continuous case, the solution $\boldsymbol{u}(t)$ of (6) can be obtained by composing the solutions $\boldsymbol{v}(t)$ on Ω_1 and $\boldsymbol{w}(t)$ on Ω_2, which satisfy the corresponding equations on the subdomains. Applying a Schwarz iteration one obtains the error equations

$$\begin{aligned} \frac{\partial \boldsymbol{d}^{k+1}}{\partial t} &= A_{(b-1)}\boldsymbol{d}^{k+1} + \boldsymbol{f}^{(e^k)} \qquad t>0\\ \boldsymbol{d}^{k+1}(0) &= \boldsymbol{0} \end{aligned} \qquad (8)$$

with $\boldsymbol{f}^{(e^k)} = (0, \dots, 0, \frac{\boldsymbol{e}^k(b-a,t)}{(\Delta x)^2})^T$ and

$$\begin{aligned} \frac{\partial \boldsymbol{e}^{k+1}}{\partial t} &= A_{(n-a)}\boldsymbol{e}^{k+1} + \boldsymbol{f}^{(d^k)} \qquad t > 0 \\ \boldsymbol{e}^{k+1}(0) &= \mathbf{0} \end{aligned} \tag{9}$$

with $\boldsymbol{f}^{(d^k)} = (\frac{\boldsymbol{d}^k(a,t)}{(\Delta x)^2}, 0, \dots, 0)^T$.

Given any vector valued function $\boldsymbol{h}(t) : \mathbb{R}^+ \longrightarrow \mathbb{R}^n$ we define

$$||\boldsymbol{h}(\cdot,\cdot)||_{\infty,\infty} := \max_{1<j<n} \sup_{t>0} |\boldsymbol{h}(j,t)|,$$

where $\boldsymbol{h}(j,t)$ denotes the j-th component of the vector $\boldsymbol{h}(t)$.

Theorem 3.1 *The Schwarz iteration for the semi-discrete heat equation with two subdomains converges at a rate depending on the size of the overlap. The error on the two subdomains decays at the rate*

$$\begin{aligned} ||\boldsymbol{d}^{2k+1}(\cdot,\cdot)||_{\infty,\infty} &\leq \left(\frac{\alpha(1-\beta)}{\beta(1-\alpha)}\right)^k ||\boldsymbol{e}^0(b-a,\cdot)||_\infty \\ ||\boldsymbol{e}^{2k+1}(\cdot,\cdot)||_{\infty,\infty} &\leq \left(\frac{\alpha(1-\beta)}{\beta(1-\alpha)}\right)^k ||\boldsymbol{d}^0(a,\cdot)||_\infty. \end{aligned}$$

Proof The proof uses the discrete maximum principle and follows as in the continuous case [GS97]. ■

The results shown for two subdomains can be generalized to an arbitrary number of subdomains, although the analysis is more involved. The theorems corresponding to Theorem 2.1 and 3.1, and their proofs, can be found in [GS97].

4 The Algorithm in the Framework of Waveform Relaxation

For a linear initial value problem

$$\frac{d\boldsymbol{u}(t)}{dt} = A\boldsymbol{u}(t) + \boldsymbol{f}(t), \quad \boldsymbol{u}(0) = \boldsymbol{u}_0$$

the standard waveform relaxation algorithm is based on a splitting of the matrix A into $A = M + N$, which yields

$$\frac{d\boldsymbol{u}(t)}{dt} = M\boldsymbol{u}(t) + N\boldsymbol{u}(t) + \boldsymbol{f}(t), \quad \boldsymbol{u}(0) = \boldsymbol{u}_0.$$

This system of ODEs is solved using an iteration of the form

$$\frac{d\boldsymbol{v}^{k+1}}{dt} = M\boldsymbol{v}^{k+1} + N\boldsymbol{v}^k + \boldsymbol{f}, \quad \boldsymbol{v}^{k+1}(0) = \boldsymbol{u}_0, \tag{10}$$

where the starting function $\boldsymbol{v}^0(t)$ is usually chosen to be constant. In the case of Block-Jacobi the matrix M is chosen to be block diagonal, for example for two subblocks

$$M = \begin{bmatrix} D_1 & \\ & D_2 \end{bmatrix}, \tag{11}$$

and N contains the remaining off diagonal blocks. This allows for solution of the subsystems D_i, $i = 1, 2$ in equation (10) in parallel. In the case where A equals $A_{(n)}$ from the semi-discrete heat equation (6), the waveform relaxation algorithm with Block-Jacobi splitting computes the same iterates as the Schwarz domain decomposition algorithm presented in subsection 3 with overlap Δx (i.e. one grid point only). This result can be generalized to an arbitrary number of subdomains, as shown in [GS97].

To extend this analogy to arbitrary overlaps, the concept of *multi-splittings* is needed, which was first introduced by O'Leary and White in [OW85] for solving large systems of linear equations on a parallel computer. Jeltsch and Pohl generalized multisplittings to linear systems of ODEs and waveform relaxation in [JP95].

Let A, M_i, N_i and E_i, $i = 1, 2$ be real $n \times n$ matrices. The set of ordered triples (M_i, N_i, E_i) for $i = 1, 2$ is called a *multi-splitting* of A if

1. $A = M_i - N_i$ for $i = 1, 2$.
2. The matrices E_l are nonnegative diagonal matrices and satisfy

$$E_1 + E_2 = I. \tag{12}$$

Using the waveform relaxation algorithm, we get two new approximations $\boldsymbol{v}_1^{k+1}$ and $\boldsymbol{v}_2^{k+1}$ at each step according to

$$\frac{d\boldsymbol{v}_i^{k+1}}{dt} = M_i \boldsymbol{v}_i^{k+1}(t) + N_i \boldsymbol{v}_i^k + \boldsymbol{f}_i, \quad \boldsymbol{v}_i^{k+1}(0) = \boldsymbol{u}_0, \ i = 1, 2 \tag{13}$$

which are combined using the matrices E_i to form a new approximation $\boldsymbol{v}^{k+1}$ by $\boldsymbol{v}^{k+1} = E_1 \boldsymbol{v}_1^{k+1} + E_2 \boldsymbol{v}_2^{k+1}$. Note that the two equations in (13) can be solved in parallel and in addition, components of $\boldsymbol{v}_i^{k+1}$ where E_i has a zero on the diagonal do not have to be computed at all provided they do not couple to other components of $\boldsymbol{v}_i^{k+1}$ where E_i has a non zero diagonal entry. Jeltsch and Pohl prove in [JP95] that the multi-splitting algorithm converges superlinearly on a finite time interval for all splittings and matrices A, and linearly on an infinite time interval if A is an M-matrix and the splitting is an M-splitting. However in the case of the semi-discrete heat equation, the rate of convergence in their analysis depends on Δx since their level of generality includes the Schwarz method with one grid point overlap and spectral radius $1 - O(\Delta x^2)$ - the block Jacobi algorithm (11). Jeltsch and Pohl also observe, on the basis of numerical experiments, that increasing the overlap accelerates the convergence rate of the algorithm. Our analysis substantiates and quantifies this observation in the specific case of the heat equation, since the E_i can be chosen in such a way that the domain decomposition algorithm described in the previous section is recovered. Choose the two splittings of A according to the two subdomains of the domain decomposition and let E_i have the value one on the diagonal in the interior of the corresponding subdomain Ω_i, including the first point of the overlap, some arbitrary distribution in the overlap satisfying (12) and zero in the interior of the other subdomain. Then the intermediate solutions $\boldsymbol{v}_i^{k+1}$ computed by the multi-splitting algorithm for the heat equation are identical to the solutions computed by the domain decomposition algorithm described in the previous section. Thus, in this case, multi-splitting gives a Δx independent rate of convergence.

Note that one could save half of the computation time by computing only even iterates on Ω_1 and odd iterates on Ω_2 or vice versa, since these two solution sequences are independent of one another. In the terminology of Domain Decomposition this would correspond to the multiplicative Schwarz algorithm with red-black ordering whereas the multi-splitting algorithm corresponds to the additive Schwarz algorithm.

The important point here is that our algorithm converges linearly, independent of the mesh size, on unbounded time intervals. Thus for certain PDEs the analysis of Jeltsch and Pohl can be refined to give Δx independent rates of convergence if sufficient overlap is used.

5 Numerical Experiments

We perform numerical experiments to measure the actual convergence rate of the algorithm. We consider first the linear example problem

$$\begin{array}{rcll} \frac{\partial u}{\partial t} & = & \frac{\partial^2 u}{\partial x^2} + 5e^{-(t-2)^2-(x-\frac{1}{4})^2} & 0 < x < 1,\ 0 < t < 3 \\ u(0,t) & = & 0 & 0 < t < 3 \\ u(1,t) & = & e^{-t} & 0 < t < 3 \\ u(x,0) & = & x^2 & 0 < x < 1. \end{array} \tag{14}$$

To solve the semi-discrete heat equation (6), (7), we use the backward Euler method in time. The experiment is done splitting the domain $\Omega = [0,1] \times [0,3)$ into the two subdomains $\Omega_1 = [0,\alpha] \times [0,3)$ and $\Omega_2 = [\beta,1] \times [0,3)$ for three pairs of values $(\alpha,\beta) \in \{(0.4,0.6),(0.45,0.55),(0.48,0.52)\}$. As initial guess for the iteration we use the constant value 1. Figure 1 shows the convergence of the algorithm at the grid point b for $\Delta x = 0.01$ and $\Delta t = 0.01$. The solid line is the predicted bound on the convergence rate according to Theorem 3.1 and the dashed line is the measured one. The measured error displayed is the difference between the numerical solution on the whole domain and the solution obtained from the domain decomposition algorithm. We also checked the robustness of the method by refining the time step and obtained similar results.

Now consider the nonlinear example problem

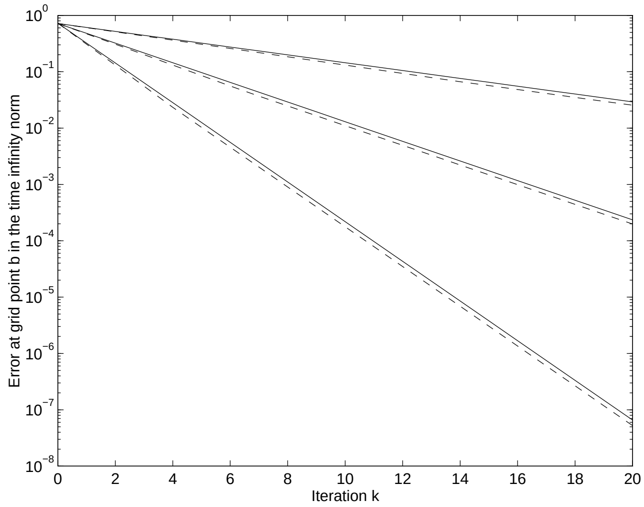
$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + 5(u - u^3) \qquad 0 < x < 1,\ 0 < t < 3 \tag{15}$$

with the same initial and boundary conditions as in the linear case. We discretize in space as before and use the backward Euler method in time for the Laplacian, keeping the nonlinear part explicit. Figure 2 shows the convergence of the algorithm at the grid point b for $\Delta x = 0.01$ and $\Delta t = 0.01$ using the same overlaps as in the linear case.

6 Conclusion

Although the analysis presented is restricted to the one-dimensional heat equation, the underlying ideas are more general. As suggested by the nonlinear example, the

Figure 1 Theoretical and measured decay rate of the error for two subdomains and three different sizes of the overlap for the linear example problem



analysis can be generalized to nonlinear problems, convection-diffusion equations, variable coefficients, and higher dimensions; this is the subject of ongoing research.

Acknowledgement

We thank Gene Golub, Olavi Nevanlinna, Morten Bjørhus and Sigitas Keras for many interesting discussions.

REFERENCES

[Bjø95] Bjørhus M. (1995) *On Domain Decomposition, Subdomain Iteration and Waveform Relaxation.* PhD dissertation, University of Trondheim, Norway, Department of Mathematical Sciences.

[Bur95] Burrage K. (1995) *Parallel and Sequential Methods for Ordinary Differential Equations.* Oxford University Press Inc.

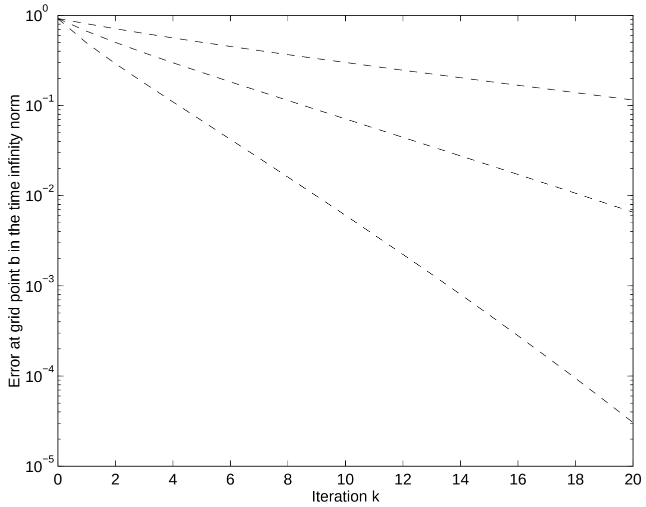
[BZ93] Bellen A. and Zennaro M. (1993) The use of Runge-Kutta formulae in waveform relaxation methods. *Appl. Numer. Math* 11: 95–114.

[Can84] Cannon J. R. (1984) *The One-Dimensional Heat Equation.* Encyclopedia of Mathematics and its Applications. Addison-Wesley.

[GK97] Giladi E. and Keller H. (1997) Space-time domain decomposition for parabolic problems. *submitted to SINUM Feb* .

[GS97] Gander M. J. and Stuart A. M. (1997) Space-time continuous analysis of waveform relaxation for the heat equation. *to appear in SIAM J. Sci. Comput.* .

Figure 2 Measured decay rate of the error for two subdomains and three different sizes of the overlap for the nonlinear example problem



[JP95] Jeltsch R. and Pohl B. (1995) Waveform relaxation with overlapping splittings. *SIAM J. Sci. Comput.* 16(1): 40–49.

[Lin94] Lindelöf E. (1894) Sur l'application des méthodes d'approximations successives á l'étude des intégrales réeles des équations différentielles ordinaires. *Journal de Mathématiques Pures et Appliqueées* 10: 117–128.

[LRSV82] Lelarasmee E., Ruehli A. E., and Sangiovanni-Vincentelli A. L. (1982) The waveform relaxation method for time-domain analysis of large scale integrated circuits. *IEEE Trans. on CAD of IC and Syst.* 1: 131–145.

[LRSW96] Lumsdaine A., Reichelt M., Squyres J., and White J. (1996) Accelerated waveform methods for parallel transient simulation of semiconductor devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 15: 716–726.

[MN87] Miekkala U. and Nevanlinna O. (1987) Convergence of dynamic iteration methods for initial value problems. *SIAM J. Sci. Stat. Comput.* 8: 459–482.

[Nev89] Nevanlinna O. (1989) Remarks on Picard-Lindelöf iterations, part i and part ii. *BIT* 29: 328–334, 535–562.

[Nev90] Nevanlinna O. (1990) Domain decomposition and iterations in parabolic problems. In *Forth Int. Symp. on Domain Decomposition Meths.*

[OW85] O'Leary D. and White R. E. (1985) Multi-splittings of matrices and parallel solution of linear systems. *SIAM J. Alg. Disc. Meth.* 6: 630–640.

[Pic93] Picard E. (1893) Sur l'application des méthodes d'approximations successives á l'étude de certaines équations différentielles ordinaires. *Journal de Mathématiques Pures et Appliqueées* 9: 217–271.

[RWA95] Reichelt M., White J., and Allen J. (1995) Optimal convolution sor acceleration of waveform relaxation with application to parallel simulation of semiconductor devices. *SIAM J. Sci. Comput.* 16(5): 1137–1158.

12

A Domain Decomposition Method for Helmholtz Scattering Problems

Souad Ghanemi

1 Introduction

We present a study of iterative nonoverlapping domain decomposition methods (DDMs) for the harmonic scattering wave equation in the 3D case. We introduce some new nonlocal transmission conditions at subdomain interfaces in order to obtain an exponential rate of convergence. This work is a natural continuation of the work by Despres [Des91]. We present numerical results for a mixed finite element approximation. The parallel performance of the method on a tightly coupled machine and a loosely connected network is also shown.

2 Domain decomposition methods

A model problem

We study the scattering scalar Helmholtz equation in three dimensions. Let $\Omega \subset \mathbb{R}^3$ be a bounded domain, Γ its boundary, and $\boldsymbol{n}$ the outgoing normal to Γ. The problem to solve is:

$$\left\{\begin{array}{lll} (a) & -\nabla(\frac{1}{\mu}\nabla u) - \omega^2 \epsilon u = f & \text{on} \quad \Omega \\ (b) & \frac{1}{\mu}\frac{\partial u}{\partial n} + i\omega\sqrt{\frac{\epsilon}{\mu}}u = 0 & \text{on} \quad \Gamma \\ (c) & u = 0 & \text{on} \quad \partial F \end{array}\right. \tag{1}$$

The boundary condition *(b)* plays an essential role and can be interpreted as a first-order absorbing boundary condition, where μ and ϵ are two positive parameters piecewise C^1. We know that for every f in $L^2(\Omega)$, (1) has a unique weak solution in $H^1(\Omega)$.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Domain decomposition methods

We apply the DDM concept to the Helmholtz scattering problem. The originality in our work is the introduction of some new nonlocal transmission conditions at the interfaces between subdomains in order to obtain an exponential rate of convergence.

Let us give a brief presentation of the method. The general idea is to split the domain Ω into several subdomains $(\Omega_k)_{k\in I}$. The solution is the limit of the following iterative process. We denote as u_k^n the restriction of the approximate solution to the domain Ω_k at step n, u_k^n being the solution of the following problem:

$$\left\{\begin{array}{lll}
\text{Find} \quad u_k^{n+1} \in H^1(\Omega_k) & & \\
\nabla(\dfrac{1}{\mu_k}\nabla u_k^{n+1}) - \omega^2\epsilon_k u_k^{n+1} = f_k & \text{in} & \Omega_k \\
\dfrac{1}{\mu_k}\dfrac{\partial u_k^{n+1}}{\partial n_k} + i\omega\sqrt{\dfrac{\epsilon_k}{\mu_k}}u_k^{n+1} = 0 & \text{on} & \Gamma_k \\
u_k^{n+1} = 0 & \text{on} & \partial F_k \\
\dfrac{1}{\mu_k}\dfrac{\partial u_k^{n+1}}{\partial n_k} + i\mathbf{T_{jk}}u_k^{n+1} = -\dfrac{1}{\mu_j}\dfrac{\partial u_j^n}{\partial n_j} + i\mathbf{T_{kj}}u_j^n = g_{kj} & \text{on} & \Sigma_{jk} \quad (*),
\end{array}\right. \tag{2}$$

where $\mathbf{T_{kj}}$ and $\mathbf{T_{jk}}$ are continuous linear operators, for which $T_{kj} = T_{jk} = T$ and

$$\mathbf{T_{kj}} : \mathbf{H^{1/2}(\Sigma_{kj})} \longrightarrow \mathbf{H^{-1/2}(\Sigma_{kj})}$$

is a symmetric isomorphism between $H^{1/2}(\Sigma_{kj})$ and $H^{-1/2}(\Sigma_{kj})$. We call equation $(*)$ on Σ_{kj} a transmission condition. The following theorem ensures that u_k^{n+1} is well defined at each step n.

Theorem 2.1 *Let $f_k \in L^2(\Omega_k)$ and $g_{kj} \in H^{-1/2}(\Sigma_{kj})$, $\mu_k \in L^\infty(\Omega_k)$ and $\epsilon_k \in L^\infty(\Omega_k)$ and piecewise C^1. Then problem (2) has an unique weak solution $u_k \in H^1(\Omega_k)$.*

Proof see [Gha96]. ■

In the following theorem, we prove that the iterative process (2) is convergent.

Theorem 2.2 *Under the hypothesis $\dfrac{1}{\mu_k}\dfrac{\partial u_k^0}{\partial n_k} \in H^{-1/2}(\partial\Omega_k)$, $\forall k \in I$, $(\mu_k, \epsilon_k) \in L^\infty(\Omega_k)^2$, piecewise C^1 and $(\dfrac{1}{\mu_k}, \dfrac{1}{\epsilon_k}) \in L^\infty(\Omega_k)^2$, the solution of equations (2), u_k^n converges in $H^1(\Omega_k)$ to u_k, the solution on Ω_k.*

Proof see [Gha96]. ■

Geometrical convergence

For the sake of clarity, we show the convergence in a homogeneous medium. The iterative process (2) is written as

$$x^{n+1} = \mathcal{A}x^n, \tag{3}$$

where x^n is the sequence defined on the interface (Γ_k, Σ_{kj}) by

$$x^n = (x^n_{\Gamma_k}, x^n_{\Sigma_{kj}}), \left\{ \begin{array}{l} x^n_{\Gamma_k} \in L^2(\Gamma_k) \\ x^n_{\Sigma_{kj}} \in L^2(\Sigma_{kj}). \end{array} \right.$$

More precisely, $V = \oplus_k \left[\oplus_{j \neq k} L^2(\Sigma_{kj}) \oplus L^2(\Gamma_k) \right]$,

$$\begin{array}{llll} \mathcal{A}: & V & \longrightarrow & V \\ & x^n = (x^n_{\Sigma_{kj}}, x^n_{\Gamma_k}) & \longrightarrow & x^{n+1} = \mathcal{A}x^n, \end{array}$$

such that:

$$\begin{aligned} x^n_{\Sigma_{kj}} &= (\mathbf{S}^*)^{-1} \frac{\partial e_k}{\partial n_k} + i\mathbf{S}e_k, \\ x^n_{\Gamma_k} &= \frac{\partial e_k}{\partial n_k} + i\omega e_k, \end{aligned}$$

where e_k is the unique solution of the problem.

$$\left\{ \begin{array}{llll} \Delta e_k + \omega^2 e_k = 0 & \text{in} & \Omega_k & (1) \\ (\mathbf{S}^*)^{-1} \dfrac{\partial e_k}{\partial n_k} + i\mathbf{S}e_k = x^n_{\Sigma_{kj}} & \text{on} & \Sigma_{kj} & (2) \\ \dfrac{\partial e_k}{\partial n_k} + i\omega e_k = x^n_{\Gamma_k} & \text{on} & \Gamma_k & (3). \end{array} \right. \tag{4}$$

x^{n+1} is constructed in the following way:

$$\begin{aligned} x^{n+1}_{\Sigma_{kj}} &= \mathcal{A}x^n_{\Sigma_{kj}} = -(\mathbf{S}^*)^{-1} \frac{\partial e_j}{\partial n_j} + i\mathbf{S}e_j, \\ x^{n+1}_{\Gamma_k} &= \mathcal{A}x^n_{\Gamma_k} = 0. \end{aligned}$$

Some properties of the $\mathcal{A}$ operator are:

- $||\mathcal{A}|| \leq 1$,
- if some eigenvalues of $\mathcal{A}$ are close to 1, then convergence is slow [GJC95].

For achieving geometrical convergence, it is necessary to use a relaxation method. The fourth $(*)$ equation in (2) is replaced by:

$$\frac{1}{\mu_k} \frac{\partial u^{n+1}_k}{\partial n_k} + iT_{kj} u^{n+1}_k = r(-\frac{1}{\mu_j} \frac{\partial u^n_j}{\partial n_j} + iT_{kj} u^n_j) + (1-r)(\frac{1}{\mu_k} \frac{\partial u^n_k}{\partial n_k} + iT_{kj} u^n_k) \;\; \textit{on} \;\; \Sigma_{kj},$$

where r is the relaxation parameter and belongs to]0,1[. As a result, we have

$$x^{n+1} = r\mathcal{A}x^n + (1-r)x^n. \tag{5}$$

Theorem 2.3 *Assume* $\dfrac{\partial u^0_k}{\partial n_k} \in H^{-1/2}(\partial \Omega_k)$, $\forall k \in I$. *If the interfaces* Σ_{kj} *do not intersect, we get an exponential rate of convergence for the relaxed iterative process:*

$$\exists \epsilon > 0 \quad \textit{such that} \quad ||(1-r)Id + r\mathcal{A}||_{\mathcal{L}(V,V)} \leq \sqrt{1 - \epsilon^2 r(1-r)} < 1. \tag{6}$$

Proof We assume the following identity:

$$\exists \epsilon > 0, \quad \forall x \in V, \quad ||(I-\mathcal{A})x||_V \geq \epsilon ||x||_V. \tag{7}$$

So, let $x \in V$ such that $||x||_V = 1$. From

$$||(I-\mathcal{A})x||_V^2 = ||\mathcal{A}x||_V^2 + ||x||_V^2 - 2Re < \mathcal{A}x, x > \geq \ (\epsilon ||x||_V)^2.$$

we deduce that

$$2Re < \mathcal{A}x, x > \leq \ 2 - \epsilon^2,$$

and

$$||(r\mathcal{A} + (1-r)I)x||_V^2 \leq (1-r)^2||x||^2 + r^2||\mathcal{A}x||^2 + 2r(1-r)Re < \mathcal{A}x, x >,$$

$$||(r\mathcal{A} + (1-r)I)x||_V^2 \leq (1-r)^2 + r^2 + 2r(1-r)(1-\epsilon^2/2) = 1 - r(1-r)\epsilon^2.$$

Finally, we have:

$$\forall x \in V, \qquad ||(r\mathcal{A} + (1-r)I)\frac{x}{||x||}||_V \leq \sqrt{1 - r(1-r)\epsilon^2}$$

$$\forall x \in V, \qquad ||(r\mathcal{A} + (1-r)I)x||_V \leq \sqrt{1 - r(1-r)\epsilon^2}||x||_V.$$

■

Assumption (7) is proved if the bijectivity of the $Id - \mathcal{A}$ operator is obtained. Then:

$$\forall x \in V, \quad x = (I-\mathcal{A})^{-1}(I-\mathcal{A})x, \quad ||x|| \leq ||(I-\mathcal{A})^{-1}||||(I-\mathcal{A})x||, \tag{8}$$

and

$$\frac{1}{||(I-\mathcal{A})^{-1}||} = \epsilon.$$

First, we show that $I - \mathcal{A}$ is injective.

If $x \in V$, is such that $x = \mathcal{A}x$, the field e solution of

$$\begin{cases} \omega^2 e_k + \Delta e_k = 0 & \text{in} \quad \Omega_k \\ \dfrac{\partial e_k}{\partial n_k} + i\omega e_k = x_k & \text{on} \quad \Gamma_k \\ (S^*)^{-1}\dfrac{\partial e_k}{\partial n_k} + iSe_k = x_{kj} & \text{on} \quad \Sigma_{kj} \end{cases}$$

satisfies

$$(S^*)^{-1}\frac{\partial e_k}{\partial n_k} + iSe_k = x_{kj} = (\mathcal{A}x)_{kj} = -(S^*)^{-1}\frac{\partial e_j}{\partial n_j} + iSe_j \quad \text{on} \quad \Sigma_{kj},$$

$$(S^*)^{-1}\frac{\partial e_j}{\partial n_j} + iSe_j = x_{jk} = (\mathcal{A}x)_{jk} = -(S^*)^{-1}\frac{\partial e_k}{\partial n_k} + iSe_k \quad \text{on} \quad \Sigma_{kj}$$

and

$$\frac{\partial e_k}{\partial n_k} + i\omega e_k = x_k = (\mathcal{A}x)_k = 0 \qquad \text{on} \qquad \Gamma_k,$$

then $e = (e_k)$ is such that $e_k \in H^1(\Omega_k)$ and

$$\begin{cases} \omega^2 e + \Delta e = 0 & \text{in} \quad \Omega - \cup\Gamma_{kj} \cup \Sigma_{kj} \\ \dfrac{\partial e_k}{\partial n_k} + i\omega e_k = 0 & \text{on} \quad \Gamma_k \\ e_k = e_j, \qquad \dfrac{\partial e_k}{\partial n_k} = -\dfrac{\partial e_j}{\partial n_j} & \text{on} \quad \Sigma_{kj}, \end{cases}$$

so, e is solution of the Helmholtz equation in Ω with $\dfrac{\partial e}{\partial n} + i\omega e = 0 \quad \text{on} \quad \partial\Omega$, and we deduce $e = 0$ and $x = 0$.

Second, we show that $I - \mathcal{A}$ is surjective.

Given a g in V where $g_{/\Gamma_k} = g_k$, $g_{/\Sigma_{jk}} = g_{kj}$, $(g_k, g_{kj}) \in (H^{-1/2}(\Gamma_k), H^{-1/2}(\Sigma_{kj}))$, we find an $x \in L^2$ such that:

$$(I - \mathcal{A})x = g,$$

which imply the existence of e belonging $H^1(\Omega)$, and satisfying the Helmholtz equation on $\Omega - \cup\Gamma_k \cup \Sigma_{kj}$ and

$$\begin{cases} \left[(S^*)^{-1}\dfrac{\partial e_k}{\partial n_k} + iSe_k\right] - \left[-(S^*)^{-1}\dfrac{\partial e_j}{\partial n_j} + iSe_j\right] = g_{kj} & \text{on} \quad \Sigma_{kj} \\ \dfrac{\partial e_k}{\partial n_k} + i\omega e_k = g_k & \text{on} \quad \Gamma_k. \end{cases} \tag{9}$$

Finally, the field e must satisfy the Helmholtz equation in each subdomain and the jump of equations (9). We note that the jump of the trace needs to belong to the space $H^{1/2}(\Sigma_{kj})$ and its normal derivative needs to belong to the space $H^{-1/2}(\Sigma_{kj})$. These conditions are satisfied if the interfaces Σ_{kj} do not intersect. We get the solution e using potential theory (see [KC93]). By defining

$$\begin{cases} x_{kj} = (S^*)^{-1}\dfrac{\partial e_k}{\partial n_k} + iSe_k & \text{on} \quad \Sigma_{kj} \\ x_k = \dfrac{\partial e_k}{\partial n_k} + i\omega e_k = g_k & \text{on} \quad \Gamma k, \end{cases} \tag{10}$$

we can easily show that $(x_k, x_{kj}) \in (L^2(\Gamma_k), L^2(\Sigma_{kj}))$. Finally, we obtain

$$(I - \mathcal{A})x = g.$$

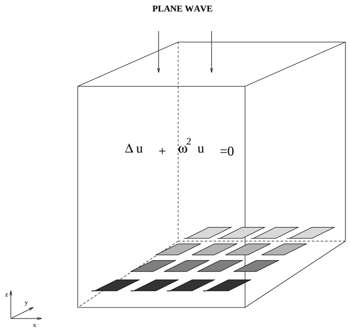
In conclusion, the geometrical convergence with a nonlocal operator is proved given a particular decomposition. The generalisation of the proof given any decomposition is an open problem.

We implement this method using the mixed hybrid finite element method [CR89]. We perform several tests to study the improvement of the convergence due to the following transparent-like operators:

$$T_p = \omega(I + \frac{c_m}{\omega^2}\Delta_{\Sigma_{kj}})^{1/2}, \tag{11}$$

where $c_m = \dfrac{1}{\sqrt{\epsilon_k.\mu_k} + \sqrt{\epsilon_j.\mu_j}}$, $\Delta_{\Sigma_{kj}}$ is the Laplace-Beltrami operator and ω is the frequency of the problem.

Figure 1 Computational problem

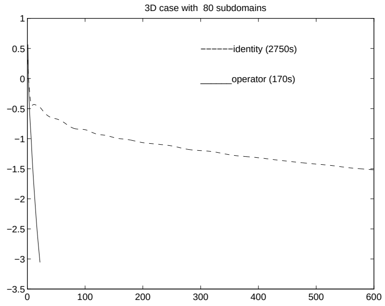


3 Numerical results

We choose a cubic domain. The scattering problem with slits located at the center of the computational domain in (3D) case is solved. The source is a plane wave (Fig. 1).

We compare the convergence of the DDM obtained with the transparent-like operator T_p and the identity operator [Des91]. Figure 2 shows the fast convergence with the new transparent transmission operator.

Figure 2 Convergence of the DDM with 80 subdomains



Our computations are done on a CRAY C90 computer. If we need 10^{-3} precision, for example, the method that does not use the nonlocal operator requires 1000 iterations and 2750 s execution time. The method with the T_p operator requires only 150 iterations and 170 s excution time. In Figure 3 on the right, the discretization step h varies for a fixed operator T_p. When the step h is fine enough (40 points per wavelength) then the convergence curves (in the log scale) confirm the exponential rate of convergence.

The same experiment is carried out with the identity operator (see Fig. 3 on the left) with mesh refinement. For fixed h, we still have exponential convergence but the corresponding rate depends on h and degenerates when h tends to 0.

Figure 3 Convergence of the DDM with identity operator and with T_p operator

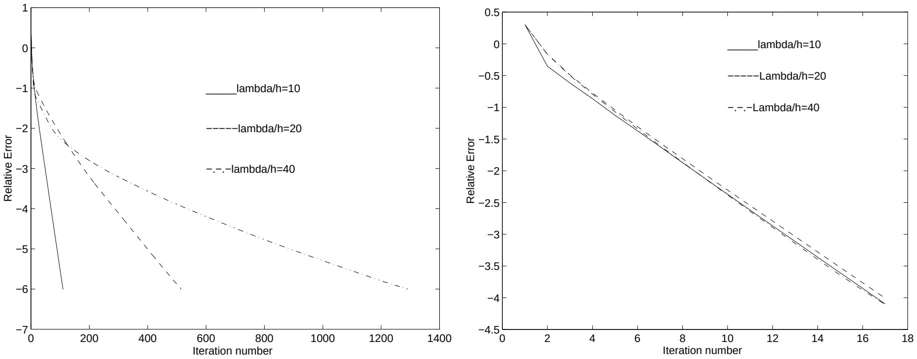


Table 1 CRAY T3D (above), SUN (below)

p	2	4	8	16
$T_s(s)$	1045	288	92	28
$T_{//}(s)$	552	103	15	2.8
$S(p)$	1.89	2.89	5.78	10.03

p	2	4	6	9	12
T_s (s)	3585	2523	465	241	167
$T_{//}$ (s)	2589	752	127	41	28
$S(p)$	1.38	2.56	3.67	5.7	5.94

4 Parallel version of the DDM

The domain decomposition algorithm can be parallelised naturally. After the discretization of the method, we have to solve several independent problems, at each step n. The solution of each subdomain can be calculated on each processor of a parallel computer. Between two iteration steps, it receives the value of both the trace and the flux of the solution, as evaluated by the processors which compute the solution of the neighbouring subdomains. The parallelization tool is PVM (Parallel Virtual Machine). Our computations are done on a heterogeneous network of workstations (SUN SPARC), and on a multiprocessor computer (CRAY T3D). The achieved speed-up $S(p)$ is defined by the ratio between the sequential execution time T_s obtained with an optimal sequential version of the method and the parallel execution time $T_{//}$ on p processors. Both tables show the various achieved speed-ups on both platforms.

We obtain a good computational performance with the CRAY T3D. The performance degrades if we increase the processor number using the heterogeneous network of workstations. Not all machines have the same computing power and it might happen that the faster machines are waiting for slower machines. This is not the case for a multiprocessor CRAY T3D, because all the processors are equivalent and the interconnecting network is optimized for parallelism.

Finally, we remark that with a parallel version, we can solve a large size problem on a distributed memory machine, which cannot solved on a sequential machine because of its memory limitations.

5 Conclusion

The conclusions of the theoretical studies are the following: we prove that the iterative process of the domain decomposition method with nonlocal transmission conditions converges. Given a particular decomposition, we obtain a geometrical convergence. The generalisation to any kind of decomposition remains an open problem.
The numerical results of the implementation of the DDM show better convergence with nonlocal transmission operators. They also show that the rate of convergence does not depend on the discretisation step. Finally, we obtain a good performance of the parallel version of the method. The communication time between processors is very small compared to the computation time.

Acknowledgement

I wish to thank my thesis advisor Patrick Joly for his inspiring guidance and Francis Collino for his helpful comments and suggestions.

REFERENCES

[CR89] Chavent G. and Roberts J. (1989) A unified physical presentation of mixed, mixed-hybrid finite elements and usual finite differences for the determination of velocities in waterflow problems. *Rapports de recherche N:1107 , INRIA, France .*

[Des91] Després B. (1991) Méthodes de décomposition de domaines pour les problèmes de propagation d'ondes en régime harmonique. *Phd thesis, Paris 9, France .*

[Gha96] Ghanemi S. (1996) Méthodes de décomposition de domaines avec conditions de transmissions non local pour les problèmes de propagation d'ondes. *Phd thesis, Paris 9, France .*

[GJC95] Ghanemi S., Joly P., and Collino F. (April 1995) Domain decomposition method for harmonic wave equations. *Third International Conference on Mathematical and Numerical Aspects of Wave Propagation, France .*

[KC93] Kreiss R. and Colton D. (1993) Inverse acoustic and electromagnetic scattering theory. *Applied mathematical sciences .*

13

Additive Schwarz, CG and Discontinuous Coefficients

I.G. Graham and M.J. Hagger

1 Introduction

This paper is concerned with the performance of the conjugate gradient(CG) method with additive Schwarz preconditioner for computing unstructured finite element approximations to the elliptic problem

$$\nabla . a \nabla u = f, \text{ on } \Omega, \;\; u = g \text{ on } \partial\Omega_D, \;\; \frac{\partial u}{\partial n} = \tilde{g} \text{ on } \partial\Omega_N. \tag{1}$$

Here $\Omega \subset \mathbb{R}^3$ is a polyhedral domain with boundary $\partial\Omega$ partitioned into disjoint subsets $\partial\Omega_D \neq \emptyset$ and $\partial\Omega_N$, each of which is composed of unions of polygons, and f, g and $\tilde{g}$ are suitably smooth given data. (Analogous results also hold in 2D.) We also assume that a is piecewise constant on each of d open disjoint polyhedral regions Λ_k, such that $\cup_{k=1}^{d} \bar{\Lambda}_k = \bar{\Omega}$, and we write $a|_{\Lambda_k} = a_k$ where each $a_k \in \mathbb{R}_+ := (0, \infty)$ is constant. We have in mind that the regions Λ_k of different material properties are fixed but may have complicated geometry and that the overall mesh used to compute u accurately will be finer than the geometry of the Λ_k. There are many applications of this type of problem, for example in groundwater flow and in electromagnetics.

After discretisation with linear finite elements on a triangulation $\mathcal{T}$ of Ω, (1) reduces to the SPD system

$$K(\mathbf{a})\mathbf{x} = \mathbf{b}(\mathbf{a}), \tag{2}$$

where the stiffness matrix and load vector depend continuously on $\mathbf{a} \in \mathbb{R}_+^{\mathrm{d}}$. Let h denote the diameter of $\mathcal{T}$ and $\mathcal{J} = \max_{k,l}\{a_k/a_l\}$. It is a standard result that $K(\mathbf{a})$ is ill-conditioned in the sense that (under suitable assumptions) $\kappa(K(\mathbf{a})) = O(h^{-2})$ as $h \to 0$ for fixed $\mathbf{a}$ and $\kappa(K(\mathbf{a})) = O(\mathcal{J})$ as $\mathcal{J} \to \infty$ for fixed h. (Here κ denotes the 2-norm condition number.) One of the striking successes of domain decomposition methods has been the construction of preconditioners for which the condition number of the preconditioned problem is bounded independently of both h and $\mathbf{a}$. We refer to these two properties as "*h-optimality*" and "*$\mathbf{a}$-optimality*" respectively. For a review

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

of the many papers on this subject, see [CM94] or [DSW96]. As far as we are aware all of these results assume meshes are obtained by structured refinement from a coarse grid (or substructures) which resolve the coefficient jumps.

In many large-scale computations, unstructured meshes are obtained by mesh generators, subdomains are obtained from mesh partitioning codes, and coarse grids are obtained by some coarsening strategy. In this context it may be difficult to implement the preconditioners covered by this theory. Recently the theory has been substantially extended to unstructured meshes - [CSZ96, CGZ96] and the references therein, and results about the h-optimality (but not the $\mathbf{a}$-optimality) of the corresponding preconditioners have been obtained. Indeed it is possible to construct some counter examples to $\mathbf{a}$-optimality when the coarse grid does not resolve the discontinuity in $\mathbf{a}$ ([GH96]). Since it seems rather unnatural to consider *unstructured* grids which are restrained to *resolve* the discontinuity in $\mathbf{a}$, this appears to suggest that unstructured grids may be bad for this type of problem. However, there is much empirical evidence to suggest the CG method remains robust to discontinuities in $\mathbf{a}$ even when they are not resolved by the discretisation and preconditioning process. In this work we prove a result which explains this phenomenon. It is obtained not by examining the condition number of the preconditioned matrix (which may be very bad) but by obtaining bounds on its eigenvalues (which, except for a small number of outliers, turn out to be very well behaved). We only have room here for a statement of our results and an idea of the proof. The necessary details are in [GH96].

2 Theoretical Results

We present here our results for the unstructured multilevel additive Schwarz preconditioner proposed in [CGZ96]. This is more general than the result in [GH96] which is about the two-level variant ([CSZ96]), but the proof of both results is identical. Let $\{\mathcal{T}^l\}_{l=0}^Q$ be a shape-regular sequence of triangulations of Ω with diameters $h^0 > h^1 > ... > h^Q = h$, where $\mathcal{T}^Q = \mathcal{T}$ is the fine mesh on which (1) is discretised. We assume in this section that this fine mesh resolves the interfaces between the regions Λ_k. We remove this restriction in §3. Assume that for each l, Ω is partitioned into non-overlapping subdomains $\tilde{\Omega}_j^l, j = 1, ..., s_l$ which are then extended to overlapping subdomains Ω_j^l with $\delta^l := \min_j \text{dist}(\partial\Omega_j^l, \partial\tilde{\Omega}_j^l) > 0$, and are such that $\partial\Omega_j^l \cap \bar{\Omega}$ contains only boundaries of tetrahedra of $\mathcal{T}^l$. More general meshes are permissible - see [CGZ96] for technical details. Let $\mathcal{N}^l$ denote the degrees of freedom in $\mathcal{T}^l$ and set $\mathcal{N}_j^l = \mathcal{N}^l \cap \Omega_j^l$. For any set of nodes $\mathcal{S}$ let $[\mathcal{S}]$ denote the space of nodal vectors on $\mathcal{S}$. Then (2) is to be solved for $\mathbf{x} \in [\mathcal{N}^Q]$. For each l and j we introduce a prolongation ${R_j^l}^T : [\mathcal{N}_j^l] \to [\mathcal{N}^Q]$ as follows: For $\mathbf{x} \in [\mathcal{N}_j^l]$, we form the piecewise linear function with value x_p at nodes $p \in \mathcal{N}_j^l$ and 0 elsewhere. Then form the piecewise linear interpolant of this with respect to $\mathcal{T}^Q$. The nodal values of this on $\mathcal{N}^Q$ are called ${R_j^l}^T\mathbf{x}$. The operator $R_j^l : [\mathcal{N}^Q] \to [\mathcal{N}_j^l]$ is defined to be the adjoint of ${R_j^l}^T$. Note that in the case $l = Q$, ${R_j^Q}^T$ is just the straightforward extension by 0 from $[\mathcal{N}_j^Q]$ to $[\mathcal{N}^Q]$. The multilevel preconditioner $M(\mathbf{a})$ is then defined by the action of its inverse:

$$M(\mathbf{a})^{-1} = \sum_{l=0}^{Q} \sum_{j=1}^{s_l} {R_j^l}^T (K(\mathbf{a})_j^l)^{-1} R_j^l, \tag{3}$$

where $K(\mathbf{a})_j^l := R_j^l K(\mathbf{a}) {R_j^l}^T$. In [CGZ96] it is proved that

$$\kappa(M(\mathbf{a})^{-1}K(\mathbf{a})) \le C(\mathbf{a}) Q^2 \max_{1 \le l \le Q} \{(h^l + h^{l-1})/\delta^l\}^2, \tag{4}$$

and so, with the appropriate choice of overlap, δ^l, we have h-optimality. The $\mathbf{a}$-optimality is an open question since the behaviour of $C(\mathbf{a})$ is unknown.

Rather than trying to analyse $C(\mathbf{a})$ which (by counterexamples in [GH96]) cannot be expected to be bounded in $\mathbf{a}$ without further assumptions, we instead obtain a bound on the number of preconditioned CG iterations needed to solve (2) as $\mathbf{a}$ varies. To do this we characterise the number of CG iterations in terms of the asymptotics of sequences of coefficients $\{\mathbf{a}^{(m)}\}_{m=1}^{\infty} \subset \mathbb{R}_+^{\mathrm{d}}$. To avoid uninteresting pathologies these are required to satisfy two mild assumptions: Firstly we require that for all k, l, $a_l^{(1)} \ge a_k^{(1)}$ implies $a_l^{(m)} \ge a_k^{(m)}$ for all $m \ge 1$; Secondly for all k, l such that $\bar{\Lambda}_k \cap \bar{\Lambda}_l \ne \emptyset$ we require that the ratio $a_k^{(m)}/a_l^{(m)}$ either approaches 0, ∞ or remains in a compact subset of $\mathbb{R}_+$ as $m \to \infty$. These assumptions are consistent with the typical applications of (1) - see [GH96].

To state our theorem, let n denote the dimension of $K(\mathbf{a})$ and let $\lambda_1^{(m)} \le \ldots \le \lambda_n^{(m)}$ denote the eigenvalues of the preconditioned matrix $M(\mathbf{a^{(m)}})^{-1}K(\mathbf{a^{(m)}})$. For any integer $0 \le L \le n-1$, set $\kappa_{L+1}^{(m)} = \lambda_n^{(m)}/\lambda_{L+1}^{(m)}$ and $\mathcal{J}^{(m)} = \max_{k,l}\{a_k^{(m)}/a_l^{(m)}\}$. Let $\mathbf{x}^j$ be the jth preconditioned CG iterate for (2), and let $\| \,.\, \|_m$ denote the energy norm induced by $K(\mathbf{a}^{(m)})$.

Theorem 1. There is an integer L and a constant C which are independent of m, h^l and δ^l such that for each $\epsilon > 0$

$$\frac{\| \mathbf{x} - \mathbf{x}^j \|_m}{\| \mathbf{x} - \mathbf{x}^0 \|_m} \le \epsilon, \text{ provided } j \ge L + \sqrt{\kappa_{L+1}^{(m)}} \left\{ \log \frac{2}{\epsilon} + L \log \frac{C}{\lambda_1^{(m)}} \right\}. \tag{5}$$

In addition $(\lambda_1^{(m)})^{-1} = O(\mathcal{J}^{(m)})$ and $\kappa_{L+1}^{(m)}$ is bounded as $\mathcal{J}^{(m)} \to \infty$, for fixed h.

Thus, apart from an additional L iterates, the number of iterates to obtain a fixed tolerance grows only logarithmically in $\mathcal{J}^{(m)}$ as $m \to \infty$. This is to be compared with the $O(\mathcal{J}^{(m)})$ growth which the condition number $\kappa_1^{(m)} = \kappa(M(\mathbf{a}^{(m)})^{-1}K(\mathbf{a}^{(m)}))$ can experience when the coarse mesh does not resolve the discontinuity in $\mathbf{a}$ [GH96].

A key question is the size of L: The answer from [GH96] is that L can never be larger than the maximal number of components of sets formed by taking unions of the Λ_k. But in many cases L is known to be smaller. The size of L depends on the *limiting form* of $\mathbf{a}^{(m)}$ as $m \to \infty$ (but not on m). A rigorous definition of L is given in [GH96]. From this it follows, for example, that if each of the Λ_k touches $\partial\Omega_D$ then $L = 0$, $\kappa_1^{(m)}$ is bounded independently of m, and the preconditioning is $\mathbf{a}$-optimal. More dramatically, suppose Ω is a square divided into a chequer-board of any number of square regions Λ_k, which are coloured alternately red and black. Suppose that

$a_k^{(m)} \to \infty$ on red squares while $a_k^{(m)} \to 0$ on black squares, then $L = 1$ at most. If any of the red squares touches $\partial\Omega_D$ then $L = 0$ and the preconditioner is again $\mathbf{a}$-optimal. We emphasise that these results do *not* require that the subdomains Ω_j^l on any of the levels have any relationship to the regions Λ_k on which a is constant.

As an example of the worst case, if $a_k^{(m)} \to \infty$ on L_0 of the regions Λ_k which do not touch $\partial\Omega_D$ and do not touch each other, and $a_k^{(m)}$ is bounded on the other regions then $L = L_0$ in the theorem.

Sketch of Proof. The proof is obtained in three stages. First it is shown that, for each k, the kth smallest eigenvalue of $M(\mathbf{a}^{(m)})^{-1}K(\mathbf{a}^{(m)})$ can be bounded below in terms of the kth smallest eigenvalue of the diagonally scaled matrix $(\text{diag}K(\mathbf{a}^{(m)}))^{-1}K(\mathbf{a}^{(m)})$ (or, its equivalent symmetric version $S(\mathbf{a}^{(m)}) := (\text{diag}K(\mathbf{a}^{(m)}))^{-1/2}K(\mathbf{a}^{(m)})(\text{diag}K(\mathbf{a}^{(m)}))^{-1/2}$). This is done by a routine application of the fact that these preconditioned matrices can be written as sums of orthogonal projections onto certain subspaces.

The second and substantial part of the proof is a characterisation of the spectrum of $S(\mathbf{a}^{(m)})$ in terms of the asymptotics of the maximum jumps of $\mathbf{a}^{(m)}$. The result is that only a fixed number L of eigenvalues of $S(\mathbf{a}^{(m)})$ may approach zero as these jumps worsen. This number depends on the geometry of the Λ_k but not on the mesh or the values of the coefficients. Some examples of the size of L have already been given above. The rest of the eigenvalues are bounded above and below by positive numbers independent of the size of the jumps. The same statement then holds for the reduced condition number $\kappa_{L+1}^{(m)}$. The third and final stage is to use a well-known extension of the convergence theory of the conjugate gradient method for the case of outlying clusters of bad eigenvalues. Full details are in [GH96].

Remark 1. An analogous estimate to (5) holds true with $\| \, . \, \|_m$ replaced by the Euclidean norm $\| \, . \, \|_2$, but inside the braces on the right-hand side of (5) we must add the term $\frac{1}{2}\log(\kappa\mathcal{J}^{(m)})$ where κ is the condition number of $K(\mathbf{1})$. This is of practical interest since the coefficient-dependent energy norm, $\| \, . \, \|_m$, is not a good place to measure the error if $a_k^{(m)} \to 0$ on some Λ_k.

Remark 2. An analogous clustering effect of preconditioners for problems of type (1) (for a more restrictive class of coefficient variations) is obtained [CNT96] and also leads to logarithmic estimates for the growth in the number of conjugate gradient iterates as the discontinuity worsens. However, the preconditioner proposed there essentially requires the solution of the global Laplace operator $a \equiv 1$ on Ω, which is implemented, for example, by embedding Ω into a rectangular grid and using fast Poisson solvers there. By contrast our present results concern standard additive Schwarz preconditioners widely used in domain decomposition methods.

3 An extension of the theory

The theory described above does not require that any of the subdomains (either at the finest level or any of the coarser levels) resolve the discontinuity in $\mathbf{a}$. However, it *does* assume that the fine mesh itself should resolve this discontinuity. This assumption simplifies the analysis of the diagonally scaled matrix, on which the theory depends, but we shall show here that it is not necessary. This generalisation has obvious practical importance since in the case of very irregular regions Λ_k, it may not even be reasonable to expect the fine grid to resolve the discontinuities.

In the interests of brevity we shall not give here a completely general extension, but instead restrict to the case of a two coefficient problem where $\Lambda_1 \subset \Omega$ and $\Lambda_2 = \Omega \backslash \bar{\Lambda}_1$. Suppose the interface $\Gamma = \bar{\Lambda}_1 \cap \bar{\Lambda}_2$ is continuous and lies entirely inside Ω. As before $a|_{\Lambda_k} = a_k \in \mathbb{R}_+$, $k = 1, 2$. Let $\mathcal{T}$ be a mesh of tetrahedra on Ω which do not need to resolve Γ. Let $\mathcal{N}$ be the nodes of $\mathcal{T}$ which are not on $\partial\Omega_D$. For $p \in \mathcal{N}$, set $\mathcal{T}(p) = \{T \in \mathcal{T} : p \in T\}$. Then for any $\mathbf{a} \in \mathbb{R}_+^2$ and $p, q \in \mathcal{N}$, $K(\mathbf{a})_{pq} = \sum_{T \in \mathcal{T}(p) \cap \mathcal{T}(q)} a_T (K_T)_{pq}$, where K_T is the piecewise linear element stiffness matrix corresponding to the Laplace operator on T and $a_T = (\int_T a)/|T|$, where $|T|$ is the volume of T. Consider the symmetrically diagonally scaled matrix $S(\mathbf{a}) = (\text{diag} K(\mathbf{a}))^{-1/2} K(\mathbf{a}) (\text{diag} K(\mathbf{a}))^{-1/2}$ and a sequence of coefficients $\mathbf{a}^{(m)}$ with constant values on the Λ_k, represented by $\{\mathbf{a}^{(m)}\} \subset \mathbb{R}_+^2$. For $T \in \mathcal{T}$ set $a_T^{(m)} = \int_T a^{(m)}/|T|$. The extreme behaviour of the spectrum of $S(\mathbf{a}^{(m)})$ can be characterised by considering the limit of $S(\mathbf{a}^{(m)})$ as $m \to \infty$. It is easy to see that $S(\mathbf{a}^{(m)})_{pq}$ is independent of m unless $p, q \in T \cap \mathcal{N}$ for some $T \in \mathcal{T}$ with $p \neq q$ and $\mathcal{T}(p) \cup \mathcal{T}(q)$ intersects both Λ_1 and Λ_2 in sets of positive volume. Then, for $T \in \mathcal{T}(p) \cup \mathcal{T}(q)$ we have $a_T^{(m)} = (|T \cap \Lambda_1|/|T|)\, a_1^{(m)} + (|T \cap \Lambda_2|/|T|)\, a_2^{(m)}$. If $\{\mathbf{a}^{(m)}\}$ is constrained to satisfy the mild assumptions of §2 then we need only consider the two cases $\lim_{m\to\infty}\{a_1^{(m)}/a_2^{(m)}\} = \infty$ or 0.

Consider $a_1^{(m)}/a_2^{(m)} \to \infty$. Introduce the slight extension of Λ_1: $\tilde{\Lambda}_1 = \cup\{T : |T \cap \Lambda_1| \neq \emptyset\}$ with extended interface $\tilde{\Gamma} = \partial\tilde{\Lambda}_1$, and the matrix $\tilde{K}_1 = \sum_{T \in \mathcal{T}} (|T \cap \Lambda_1|/|T|) K_T$, which corresponds to a Neumann problem on $\tilde{\Lambda}_1$ for an operator of the form (1) with coefficient $|T \cap \Lambda_k|/|T|$ on each T. Let $\tilde{K}_2$ be the finite element matrix on $\Omega \backslash \tilde{\Lambda}_1$ corresponding to Dirichlet condition on $\tilde{\Gamma}$ and given boundary conditions on $\partial\Omega$. Then $\lim_{m\to\infty} S(\mathbf{a}^{(m)}) \to \tilde{S}$, where $\tilde{S}$ is the diagonally scaled version of the block diagonal matrix $\text{diag}(\tilde{K}_1, \tilde{K}_2)$. $\tilde{S}$ has a single zero eigenvalue with all other eigenvalues positive (but depending on h). So $S(\mathbf{a}^{(m)})$ has a single eigenvalue approaching zero as $m \to \infty$ and Theorem 1 holds with $L = 1$. If $a_1^{(m)}/a_2^{(m)} \to 0$, then $S(\mathbf{a}^{(m)})$ also approaches a diagonally scaled version of a matrix of the general form $\text{diag}(\tilde{K}_1, \tilde{K}_2)$. But here $\tilde{K}_2$ is a stiffness matrix on a slight extension, $\tilde{\Lambda}_2$, of Λ_2 with Neumann condition on $\partial\tilde{\Lambda}_2 \backslash \partial\Omega$ and given mixed conditions on $\partial\Omega$. $\tilde{K}_1$ is a matrix corresponding to a Dirichlet problem on $\Omega \backslash \tilde{\Lambda}_2$. This time $\tilde{S}$ has all positive eigenvalues and Theorem 1 holds with $L = 0$.

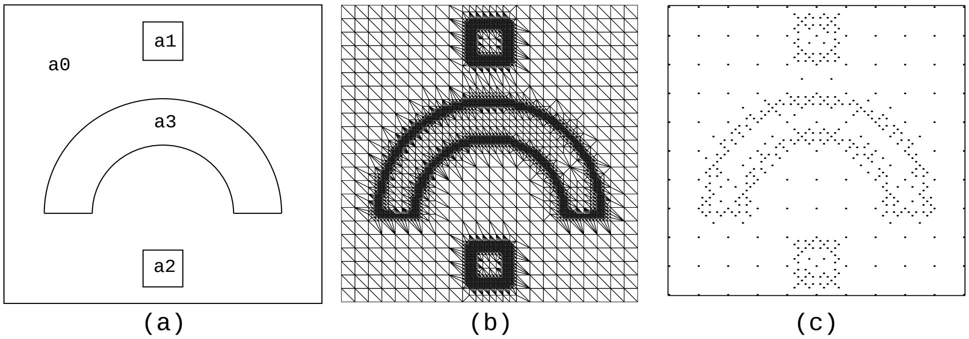
An extension of this argument to many coefficients will lead to the proof of Theorem 1, in the case when the fine grid does not resolve the discontinuity. The extension to $\| \cdot \|_2$ mentioned in Remark 1 also holds by the arguments in [GH96].

4 A Numerical Example

To demonstrate our results we consider a two-dimensional problem with geometry motivated by electromagnetic field computations, see [EST94] and the references contained therein. Here there are three interior regions with varying material properties, as in Figure 1(a). The domain is the unit square and the regions with differing material properties are given by $\Lambda_1 = (0.44, 0.56) \times (0.81, 0.94)$, $\Lambda_2 =$

$(0.44, 0.56) \times (0.063, 0.19), \Lambda_3 = \{(x,y) : 0.063 \leq (x-0.5)^2+(y-0.3)^2 \leq 0.14, y \geq 0.3\}$. For the static field case the differential equation is in the form (1), and for this experiment we imposed homogeneous Neumann boundary conditions on the left and right boundaries and Dirichlet boundary conditions of 1 and 0 to the top and bottom boundaries respectively of the unit square.

Figure 1 Geometry of an electromagnetic problem (a), fine mesh (b) and coarse mesh node points (c)



The domain is discretised with a uniform mesh of triangles except for strong refinement near the boundaries of the regions with differing material properties. This results in a mesh of 30856 triangles with 15484 nodes (Figure 1(b)). These do not resolve completely the semicircular geometry of Λ_3.

In this experiment we use the two-level additive Schwarz method, see for example [CSZ96], for which we, in contrast to (3), require the solution of a global coarse problem together with local problems on subdomains of the fine mesh. In principle the coarse mesh is not required to have any direct relation to the fine mesh. However, it may be expected that a coarse mesh which pays no attention to the underlying PDE (e.g. fails to have some refinement where the fine mesh is refined in this example) may not work well. To determine our coarse mesh we first impose a uniform coarse mesh and then perform hierarchical local refinement with "slave nodes" as, e.g., in §7 of [CM94], to increase the density in regions where the fine mesh is dense. The result has 465 nodes and is pictured in Figure 1(c). This coarse mesh is represented by a locally uniform data structure which is completely uniform in large sections of the domain, this allows for a very simple and efficient implementation. More details on the creation and performance of such coarse meshes will be available in a future publication. The partitioning of the fine mesh into the local subdomains is performed using the graph partitioning package METIS[KK95]. Use of this package allows us to produce load balanced connected subdomains, with a single node overlap, based only on the connectivity of the mesh. Hence the geometry of the problem has no direct bearing on the subdomains and neither the fine mesh, coarse mesh nor subdomains resolve the discontinuity in **a**.

To demonstrate the effect of the discontinuous coefficients on the additive Schwarz preconditioner we use five sets of coefficient values, as specified in Table 1. Each of these problems is also tested with a range of subdomain numbers, from 15 to 120. Three different preconditioners are tested in each case: One with no coarse solve (AS),

Problem	a0	a1	a2	a3
1	1.0	1.0	1.0	1.0
2	1.0	1.0(-1)	1.0(1)	1.0(2)
3	1.0	1.0(-2)	1.0(2)	1.0(4)
4	1.0	1.0(-3)	1.0(3)	1.0(6)
5	1.0	1.0(-4)	1.0(4)	1.0(8)

Table 1 Coefficient values for the test problems

15 Subdomains				30 Subdomains			
Problem	AS	ASC*	ASC	Problem	AS	ASC*	ASC
1	70	25	27	1	87	26	28
2	92	27	26	2	116	31	28
3	109	30	29	3	131	35	33
4	117	31	31	4	149	36	35
5	135	31	31	5	165	36	34
60 Subdomains				120 Subdomains			
Problem	AS	ASC*	ASC	Problem	AS	ASC*	ASC
1	123	33	33	1	158	39	34
2	150	36	32	2	186	40	36
3	172	38	37	3	213	44	39
4	196	39	37	4	241	45	43
5	220	39	37	5	272	45	44

Table 2 Results for the 15,30,60 and 120 subdomain case

a second with a coarse solve but based on a purely uniform coarse mesh, with 121 nodes, (ASC*), and the third with coarse mesh pictured in Figure 1(c) (ASC). An initial guess of **0** was used for the CG algorithm.

Table 2 shows the number of iterations of the preconditioned CG method required to satisfy the convergence criterion $\| \mathbf{x}^j - \mathbf{x} \|_2 / \| \mathbf{x} \|_2 < 10^{-5}$, where $\mathbf{x}^j$ is the jth iterate and $\mathbf{x}$ is the true solution computed using an exact factorisation. In all cases convergence to the same tolerance in the energy norm took place within one or two iterates of the results given here. The results for the preconditioner with no coarse solve (AS) show the expected logarithmic growth in the number of CG iterations required, as the jumps in the coefficient worsen. Additionally an increase in the number of subdomains also results in the expected increase in iterations. For the remaining two preconditioners (ASC* and ASC) the growth with respect to the number of subdomains is almost identical, indicating that, for this problem, a simple coarse mesh would be sufficient. The results for ASC* in this case raise the interesting question of whether a uniform coarse mesh would suffice for more general preconditioning tasks. Preliminary experiments indicate that this is not always the case. This will be the subject of future work.

Acknowledgement

This work was supported by UK EPSRC grant GR/J88616.

REFERENCES

[CGZ96] Chan T. F., Go S., and Zou J. (1996) Multilevel domain decomposition and multigrid methods for unstructured meshes: Algorithms and theory. In Glowinski R., Périaux J., Shi Z.-C., and Widlund O. B. (eds) *Proceedings of the Eighth International Conference on Domain Decomposition.* Wiley and Sons, Chichester.

[CM94] Chan T. F. and Mathew T. P. (1994) Domain decomposition algorithms. In Iserles A. (ed) *Acta Numerica*, pages 61–143. Cambridge University Press.

[CNT96] Cai X., Nielsen B. F., and Tveito A. (1996) An analysis of a preconditioner for the discretised pressure equation arising in reservoir simulation. Preprint.

[CSZ96] Chan T. F., Smith B., and Zou J. (1996) Overlapping schwarz methods on unstructured meshes using non-matching coarse grids. *Numer. Math.* 73: 149–167.

[DSW96] Dryja M., Sarkis M. V., and Widlund O. B. (1996) Multilevel Schwarz methods for elliptic problems with discontinuous coefficients in 3 dimensions. *Numer. Math.* 72(3): 313–348.

[EST94] Emson C. R. I., Simkin J., and Trowbridge C. W. (1994) A status report on electromagnetic field computation. *IEEE Trans. Magnetics* 30(4): 1533–1540.

[GH96] Graham I. G. and Hagger M. J. (1996) Unstructured additive Schwarz - CG method for elliptic problems with highly discontinuous coefficients. Submitted to SIAM J. Sci. Comp., June 1996.

[KK95] Karypis G. and Kumar V. (1995) *METIS: Unstructured graph partitioning and sparse matrix ordering system.* Dept. Computer Science, Univ. of Minnesota, Minneapolis.

14

A Stable Spectral Multi-Domain Method for the Unsteady, Compressible Navier-Stokes Equations

J. S. Hesthaven

1 Introduction

In this paper we develop a multi-domain scheme, based on quadrilaterals as the building-block, for stable approximation of the two-dimensional compressible Navier-Stokes equations on conservation form. Although the presentation here is self-contained, the results rely heavily on a series of recent papers [HG96, Hes97a, Hes97b], to which we also refer for proofs and theoretical details. For ease of exposure, we have chosen to restrict the attention to schemes for two-dimensional subsonic flows. Details for supersonic flows and three-dimensional schemes can be found in [Hes97b].

Previous work on spectral multi-domain methods for the compressible Navier-Stokes equations is rather sparse. Only recently have several methods appeared [Kop93, Han93, KK96] with the emphasis being on methods for steady state problems. All previous methods for viscous flows are based on a treatment of the inviscid part of the equation, in most cases by applying methods known from the Euler equations, and a separate treatment of the viscous part of the equation. This second contribution is then applied as a correction to the result obtained from the inviscid patching.

The main difference between previously proposed methods and the one introduced here is that we develop a patching scheme which accounts for the inviscid and viscous part of the equation simultaneously. This approach is made possible by implementing the interface conditions using a penalty term [FG88], hence allowing for boundary conditions of a general type.

In Section 2 we introduce some background and notation. Section 3 introduces the complete scheme and theorems for well-posedness in a general plane domain and asymptotic stability of the scheme in a curvilinear quadrilateral. An example of the performance of the scheme for a non-trivial test case is presented in Section 4, which also contains a few concluding remarks.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 General Background and Notation

We wish to devise a scheme for approximating wave dominated problems in the domain, $\Omega \subset \mathsf{R}^2$, enclosed by the boundary $\delta\Omega$. To obtain such solutions we employ polynomial expansions to approximate the unknown functions and their spatial derivatives. As is well known, the most natural and computationally efficient way of applying polynomial expansions in several dimensions is through the use of tensor products. This procedure, however, requires that the computational domain is diffeomorphic to the unit square. To surmount this limitation, we construct Ω using K non-overlapping general quadrilaterals, $\mathsf{D}^k \subset \mathsf{R}^2$, such that $\Omega = \bigcup_{k=1}^K \mathsf{D}^k$. In what remains the emphasis will be on schemes for addressing problems in D^k and for simplicity we will by D with boundary $\delta\mathsf{D}$, refer to any quadrilateral domain unless clarification is deemed necessary.

To apply the tensor product formulation we require that there exists a diffeomorphism, $\Psi : \mathsf{D} \to \mathsf{I}$, where $\mathsf{I} \subset \mathsf{R}^2$ is the unit square, i.e., $\mathsf{I} \in [-1,1]^2$. We will return to the specification of the map, Ψ, shortly. For convenience, we term the coordinates, $\boldsymbol{x} \in \mathsf{D}$, as (x, y) and (x_1, x_2) interchangeably. Likewise, we introduce the coordinates, $\boldsymbol{\xi} \in \mathsf{I}$, named (ξ, η).

As mentioned briefly, the map, $\Psi : \mathsf{D} \to \mathsf{I}$, plays an important role in the application of polynomial methods to problems in general geometries. To establish a one to one correspondence between the unit square and the general quadrilateral we construct the global map using transfinite blending functions as originally suggested in [GH73]. We refer to [Hes97b] for a thorough account of this procedure within the present context.

Once the global map, Ψ, has been constructed, we compute the metric of the mapping, the corresponding transformation Jacobian and outward pointing normal vectors at all points of the enclosing edges of the quadrilateral. Spatial derivatives are obtained through the chain rule and the relevant operators are all expressed in general curvilinear coordinates.

Approximation in I is done by a standard pseudospectral method using tensor products of interpolating Lagrange polynomials based on the Gauss-Lobatto nodal sets of Jacobi polynomials. We refer to [Fun92] for a general discussion of these techniques and to [HG96, Hes97a, Hes97b] for a thorough discussion within the present context.

3 A Stable Scheme for Navier-Stokes Equations

Consider the non-dimensional, compressible Navier-Stokes equations on conservation form

$$\frac{\partial \boldsymbol{q}}{\partial t} + \nabla \cdot \boldsymbol{\Pi} = \frac{1}{\mathrm{Re}_{\mathrm{ref}}} \nabla \cdot \boldsymbol{\Pi}_\nu \quad , \tag{1}$$

where we introduce the state vector, $\boldsymbol{q} = [\rho, \rho u, \rho v, E]^T$, and the inviscid flux tensor, $\boldsymbol{\Pi} = (\boldsymbol{F}_1, \boldsymbol{F}_2)$, with the elements $\boldsymbol{F}_1 = [\rho u, \rho u^2 + p, \rho uv, (E+p)u]^T$ and likewise $\boldsymbol{F}_2 = [\rho v, \rho uv, \rho v^2 + p, (E+p)v]^T$. Here ρ is the density, $\boldsymbol{u} = (u, v)$ is the Cartesian velocity, E is the total energy and p is the pressure.

The total energy, $E = \rho T + \frac{1}{2}\rho \boldsymbol{u} \cdot \boldsymbol{u}$, and the pressure are assumed to be related through the ideal gas law, $p = (\gamma - 1)\rho T$, where T is the temperature field and

$\gamma = c_p/c_v$ is the ratio between the heat capacities at constant pressure (c_p) and volume (c_v), respectively, and is assumed constant.

The elements of the viscous flux tensor, $\boldsymbol{\Pi}_\nu = (\boldsymbol{F}_1^\nu, \boldsymbol{F}_2^\nu)$, are given as $\boldsymbol{F}_1^\nu = [0, \tau_{xx}, \tau_{yx}, \tau_{xx}u + \tau_{yx}v + \frac{\gamma k}{\mathrm{Pr}}\frac{\partial T}{\partial x}]^T$ and also $\boldsymbol{F}_2^\nu = [0, \tau_{xy}, \tau_{yy}, \tau_{xy}u + \tau_{yy}v + \frac{\gamma k}{\mathrm{Pr}}\frac{\partial T}{\partial y}]^T$. Considering only Newtonian fluids, the stress tensor elements are

$$\tau_{x_i x_j} = \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + \delta_{ij}\lambda\sum_{k=1}^{2}\frac{\partial u_k}{\partial x_k} \quad ,$$

where δ_{ij} is the Kronecker delta-function and $(u_1, u_2) = \boldsymbol{u}$. Here μ is the dynamic viscosity, λ is the bulk viscosity and k is the coefficient of thermal conductivity.

The equations are normalized using the reference values, $u_{\mathrm{ref}} = u_0$, $\rho_{\mathrm{ref}} = \rho_0$, $p_{\mathrm{ref}} = \rho_0 u_0^2$, $T_{\mathrm{ref}} = u_0^2/c_v$ and a reference length L, where (ρ_0, u_0) is a given characteristic state. This yields a Reynolds number as $\mathrm{Re} = \rho_0 u_0 L/\mu_0$ and a Prandtl number as $\mathrm{Pr} = c_p\mu_0/k_0$. Note, that the Reynolds number in Eq.(1), $\mathrm{Re}_{\mathrm{ref}}$, based on the reference values, in general is different from Re. In the remaining part of the paper we shall refer to the latter as the Reynolds number. With this normalization we need to specify the Mach number, M, the Reynolds number, Re, the length scale, L, and a dimensional temperature, T_0.

We consider only atmospheric air and take $\gamma = 1.4$ and $\mathrm{Pr} = 0.72$. To model the temperature dependence of the dynamic viscosity we use Sutherland's viscosity law [Sch79]. Assuming that the Prandtl number is constant allows for modeling the temperature dependency of the coefficient of thermal conductivity similarly and we adopt Stokes hypothesis (see e.g. [Sch79]) to obtain $\lambda = -\frac{2}{3}\mu$ in all simulations.

Well-posed Patching Conditions

To derive a set of well-posed boundary conditions for the compressible Navier-Stokes equations on a general plane surface, we introduce the transformation derivatives

$$\mathcal{A}_i = \frac{\partial \boldsymbol{F}_i}{\partial \boldsymbol{q}} \quad \text{and} \quad \mathcal{B}_{ij} = \frac{1}{2}\left(\frac{\partial \boldsymbol{F}_i^\nu}{\partial \boldsymbol{q}_{x_j}} + \frac{\partial \boldsymbol{F}_j^\nu}{\partial \boldsymbol{q}_{x_i}}\right) \quad ,$$

where $\boldsymbol{q}_{x_i} = \partial\boldsymbol{q}/\partial x_i$. To arrive at the proper boundary operator, we find it convenient also to introduce the operators

$$\mathcal{A} = \sum_{i=1}^{2}\mathcal{A}_i n_i \quad \text{and} \quad \mathcal{B}_{x_i} = \sum_{j=1}^{2}\mathcal{B}_{ij} n_j \quad ,$$

where $\boldsymbol{n} = (n_1, n_2) = |\boldsymbol{n}|\hat{\boldsymbol{n}}$ is an outward pointing normal vector at $\delta\mathsf{D}$ of length $|\boldsymbol{n}|$.

Provided the solution, $\boldsymbol{q}$, is smooth it is sufficient to consider well-posedness and stability of the linearized and localized set of equations as discussed in [KL89] and applied extensively in [HG96, Hes97a, Hes97b].

A diagonalizing similarity transformation for arbitrary $\boldsymbol{n}$ for the constant coefficient operator, $\mathcal{A}(\boldsymbol{q}_0)$, was given by Warming et al. [WBH75]. Applying this transformation yields the diagonal matrix, $\mathcal{A}^{\mathbf{n}} = (\mathcal{S}^{\mathbf{n}})^{-1}\mathcal{A}\mathcal{S}^{\mathbf{n}}$, with the diagonal elements

$$\lambda_1^{\mathbf{n}} = \boldsymbol{u}_0\cdot\boldsymbol{n} + c_0|\boldsymbol{n}| \quad , \quad \lambda_2^{\mathbf{n}} = \lambda_3^{\mathbf{n}} = \boldsymbol{u}_0\cdot\boldsymbol{n} \quad , \quad \lambda_4^{\mathbf{n}} = \boldsymbol{u}_0\cdot\boldsymbol{n} - c_0|\boldsymbol{n}| \quad ,$$

representing the advective velocities of the characteristic functions, $\boldsymbol{R}^{\mathbf{n}} = (\mathcal{S}^{\mathbf{n}})^{-1}\boldsymbol{q}$, along the direction given by $\boldsymbol{n}$, given as

$$\boldsymbol{R}^{\mathbf{n}} = \left[\begin{array}{c} \boldsymbol{m}\cdot\hat{\boldsymbol{n}} + \frac{\gamma-1}{c_0}\left(E + \frac{1}{2}\rho q_0^2 - \rho\boldsymbol{u}_0\cdot\boldsymbol{u}\right) \\ \rho - \frac{\gamma-1}{c_0^2}\left(E + \frac{1}{2}\rho q_0^2 - \rho\boldsymbol{u}_0\cdot\boldsymbol{u}\right) \\ \boldsymbol{m}\cdot\hat{\boldsymbol{k}} \\ -\boldsymbol{m}\cdot\hat{\boldsymbol{n}} + \frac{\gamma-1}{c_0}\left(E + \frac{1}{2}\rho q_0^2 - \rho\boldsymbol{u}_0\cdot\boldsymbol{u}\right) \end{array} \right] ,$$

where we introduce the linearized momentum, $\boldsymbol{m} = \rho(\boldsymbol{u} - \boldsymbol{u}_0)$ and the tangential vector, $\boldsymbol{k} = |\boldsymbol{n}|\hat{\boldsymbol{k}} = |\boldsymbol{n}|(-\hat{n}_2, \hat{n}_1)$. Here $c_0 = \sqrt{\gamma p_0/\rho_0}$ represents the sound speed at the linearizing state.

Likewise, we also define the transformed viscous matrices, $\mathcal{B}^{\mathbf{n}}_{x_i} = (\mathcal{S}^{\mathbf{n}})^{-1}\mathcal{B}_{x_i}\mathcal{S}^{\mathbf{n}}$, to finally obtain the viscous correction vector

$$\boldsymbol{G}^{\mathbf{n}} = \mathcal{B}^{\mathbf{n}}_x\frac{\partial\boldsymbol{R}^{\mathbf{n}}}{\partial x} + \mathcal{B}^{\mathbf{n}}_y\frac{\partial\boldsymbol{R}^{\mathbf{n}}}{\partial y} = |\boldsymbol{n}| \left[\begin{array}{c} \frac{k_0(\gamma-1)}{2\mathrm{Pr}\rho_0}\boldsymbol{V}_1\cdot\hat{\boldsymbol{n}} + \frac{\lambda_0+2\mu_0}{2\rho_0}\boldsymbol{V}_2\cdot\hat{\boldsymbol{n}} - \frac{\lambda_0+\mu_0}{2\rho_0}\boldsymbol{V}_3\cdot\hat{\boldsymbol{k}} \\ -\frac{k_0(\gamma-1)}{2\mathrm{Pr}c_0}\boldsymbol{V}_1\cdot\hat{\boldsymbol{n}} \\ -\frac{\mu_0}{\rho_0}\boldsymbol{V}_3\cdot\hat{\boldsymbol{n}} + \frac{\lambda_0+\mu_0}{4\rho_0}\boldsymbol{V}_2\cdot\hat{\boldsymbol{k}} \\ \frac{k_0(\gamma-1)}{2\mathrm{Pr}\rho_0}\boldsymbol{V}_1\cdot\hat{\boldsymbol{n}} - \frac{\lambda_0+2\mu_0}{2\rho_0}\boldsymbol{V}_2\cdot\hat{\boldsymbol{n}} + \frac{\lambda_0+\mu_0}{2\rho_0}\boldsymbol{V}_3\cdot\hat{\boldsymbol{k}} \end{array} \right] .$$

Here we have, for simplicity, introduced the vectors

$$\boldsymbol{V}_1 = \nabla R^{\mathbf{n}}_1 + \nabla R^{\mathbf{n}}_4 - \frac{2c_0}{(\gamma-1)}\nabla R^{\mathbf{n}}_2 \ , \ \ \boldsymbol{V}_2 = \nabla R^{\mathbf{n}}_1 - \nabla R^{\mathbf{n}}_4 \ , \ \ \boldsymbol{V}_3 = \nabla R^{\mathbf{n}}_3 \ ,$$

where $\boldsymbol{V}_1$ accounts for the normal heat flux, $\boldsymbol{V}_2$ for the normal stress and $\boldsymbol{V}_3$ for the effects of the tangential stress.

We are now in a position to state the following

Theorem 14.1 *Assume there exists a solution, $\boldsymbol{q}$, to the compressible Navier-Stokes equations on a general plane surface, D, enclosed by an almost smooth boundary, $\delta\mathsf{D}$, with the outward pointing normal vector, $\boldsymbol{n}$, uniquely defined at all points with the exception of a finite number of sets having measure zero in R.*

Assume also that the fluid properties are constrained as

$$\mu \geq 0 \ , \ \ \lambda \leq 0 \ , \ \ \mu + \lambda \geq 0 \ , \ \ \frac{\gamma k}{\mathrm{Pr}} \geq 0 \ , \ \ \gamma \geq 1 \ .$$

Provided the boundary operator is constructed such that

$$\forall\boldsymbol{x} \in \delta\mathsf{D}, \forall i \in [1,4] : \ \ R^{\mathbf{n}}_i\left[-\frac{1}{2}\lambda^{\mathbf{n}}_i R^{\mathbf{n}}_i + \frac{1}{\mathrm{Re}_{\mathrm{ref}}}G^{\mathbf{n}}_i\right] \leq 0 \ ,$$

where $R^{\mathbf{n}}_i$ and $G^{\mathbf{n}}_i$ represents the components of the vectors, $\boldsymbol{R}^{\mathbf{n}}$ and $\boldsymbol{G}^{\mathbf{n}}$, respectively, the constant coefficient problem is well-posed.

From this result it is straightforward to obtain a set of maximal dissipative boundary conditions of the form

$$\mathcal{R}^{\mathbf{n}}_{\pm}\boldsymbol{R}^{\mathbf{n}} + \frac{1}{\mathrm{Re}_{\mathrm{ref}}}\mathcal{G}_{\pm}\boldsymbol{G}^{\mathbf{n}} = 0 \ ,$$

where the subscript ± refers to the situations for which the boundary is an inflow, $\boldsymbol{u}_0 \cdot \boldsymbol{n} < 0$, or an outflow, $\boldsymbol{u}_0 \cdot \boldsymbol{n} > 0$, boundary and we introduce the four matrices, $\mathcal{R}^{\mathbf{n}}_{\pm}$ and $\mathcal{G}_{\pm}$, to construct the appropriate boundary operator.

For the subsonic inflow case, well-posedness appears for $\mathcal{R}^{\mathbf{n}}_{-} = \mathrm{diag}\,(0, |\lambda^{\mathbf{n}}_2|, |\lambda^{\mathbf{n}}_3|, |\lambda^{\mathbf{n}}_4|)$ and $\mathcal{G}_{-} = \mathrm{diag}(1,1,1,1)$. Likewise, for the subsonic outflow case we obtain the operator as $\mathcal{R}^{\mathbf{n}}_{+} = \mathrm{diag}\,(0,0,0,|\lambda^{\mathbf{n}}_4|)$ and $\mathcal{G}_{+} = \mathrm{diag}(0,1,1,1)$. The matrices corresponding to supersonic inflow and outflow are given in [Hes97b].

The singular nature of $\mathcal{G}_{+}$ is a consequence of the fact that for $G^{\mathbf{n}}_2 = 0$ we obtain that $G^{\mathbf{n}}_1 = -G^{\mathbf{n}}_4$. Consequently, only three conditions are required at outflow.

Similar to what was discussed in [HG96], we observe that the number of necessary boundary conditions at inflow (4) and outflow (3) conforms with results reported in [Str77]. We also recall that the boundary operator remains well-posed even in the case where the Reynolds number approaches infinity and we obtain the characteristic boundary conditions for the inviscid, compressible Euler equations.

The Stable Semi-Discrete Scheme

Establishing the boundary operator leading to a well-posed problem when considering the solution of the compressible Navier-Stokes equations in a general domain, allows us to develop an asymptotically stable scheme for approximating the equations in a general curvilinear domain. Although a similar approach may be applied for constructing schemes in general domains, we restrict the attention to the quadrilateral domain.

We propose to solve the compressible Navier-Stokes equations in a quadrilateral using a collocation method as

$$\begin{aligned}\frac{\partial \boldsymbol{q}}{\partial t} + \nabla \cdot \boldsymbol{\Pi} = \quad & \frac{1}{\mathrm{Re}_{\mathrm{ref}}} \nabla \cdot \boldsymbol{\Pi}_{\nu} \\ & -\tau Q(\boldsymbol{x}) \mathcal{S}^{\mathbf{n}} \left[\mathcal{R}^{\mathbf{n}}_{\pm} \left(\boldsymbol{R}^{\mathbf{n}} - \boldsymbol{R}^{\mathbf{n}}_{BC} \right) + \frac{1}{\mathrm{Re}_{\mathrm{ref}}} \mathcal{G}_{\pm} \left(\boldsymbol{G}^{\mathbf{n}} - \boldsymbol{G}^{\mathbf{n}}_{BC} \right) \right] \quad , \end{aligned} \tag{2}$$

where we introduce $\boldsymbol{R}^{\mathbf{n}}_{BC}$ and $\boldsymbol{G}^{\mathbf{n}}_{BC}$ to account for the boundary conditions in characteristic form at the various boundaries, be they sub-domain boundaries or open boundaries. The matrix, $\mathcal{S}^{\mathbf{n}}$, coming from the similarity transform of $\mathcal{A}$ along $\boldsymbol{n}$, is given as

$$\mathcal{S}^{\mathbf{n}} = \begin{bmatrix} \alpha & 1 & 0 & \alpha \\ \alpha(u + c\hat{n}_1) & u & -\hat{n}_2 & \alpha(u - c\hat{n}_1) \\ \alpha(v + c\hat{n}_2) & v & \hat{n}_1 & \alpha(v - c\hat{n}_2) \\ \alpha(H + c\boldsymbol{u} \cdot \hat{\boldsymbol{n}}) & \frac{1}{2}\boldsymbol{u} \cdot \boldsymbol{u} & \boldsymbol{u} \cdot \hat{\boldsymbol{k}} & \alpha(H - c\boldsymbol{u} \cdot \hat{\boldsymbol{n}}) \end{bmatrix} \quad ,$$

where we have the constant, $\alpha = 1/(2c)$, and the specific stagnation enthalpy, $H = (E + p)/\rho$. In $\mathcal{S}^{\mathbf{n}}$, all physical variables refer to the state, $\boldsymbol{q}_0$, around which we have linearized. The function $Q(\boldsymbol{x})$ is defined as

$$Q(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in \delta \mathsf{D} \\ 0 & \text{otherwise} \end{cases} \quad ,$$

ensuring that Eq.(2) is modified at the boundaries only.

The conditions on τ ensuring asymptotic stability are given in the following Theorem.

Theorem 14.2 *Assume there exists a solution, $\boldsymbol{q}$, to the compressible Navier-Stokes equations in a general curvilinear quadrilateral domain, D, enclosed by the almost smooth boundary, $\delta\mathsf{D}$ and that the flow is purely subsonic. Assume also that there exists an diffeomorphism, $\Psi : \mathsf{D} \to \mathsf{I}$, which maps D onto the unit square, I.*

The fluid properties must by constrained as

$$\mu \geq 0\ ,\ \ \lambda \leq 0\ ,\ \ \mu + \lambda \geq 0\ ,\ \ \frac{\gamma k}{\mathrm{Pr}} \geq 0\ ,\ \ \gamma \geq 1\ \ .$$

Let

$$\kappa = \frac{1}{\mathrm{Re}_{\mathrm{ref}}} \frac{|\boldsymbol{n}|^2}{2\tilde{\omega}} \frac{1}{\rho_0 |\boldsymbol{u}_0 \cdot \boldsymbol{n}|} \max\left(\mu_0, 2\mu_0 + \lambda_0, \frac{\gamma k_0}{\mathrm{Pr}}\right) \ \ .$$

Approximating the solution of the linearized constant coefficient version of Eq.(2) using a collocation method yields an asymptotically stable scheme provided the penalty parameters are bounded as

$$\frac{1}{\tilde{\omega}\kappa}\left(1 + \kappa - \sqrt{1+\kappa}\right) \leq \tau \leq \frac{1}{\tilde{\omega}\kappa}\left(1 + \kappa + \sqrt{1+\kappa}\right) \ \ .$$

The correct choice of $\tilde{\omega}$ and the outward pointing normal vector, $\boldsymbol{n}$, depending on whether an edge or a vertex, is considered is given in the Appendix.

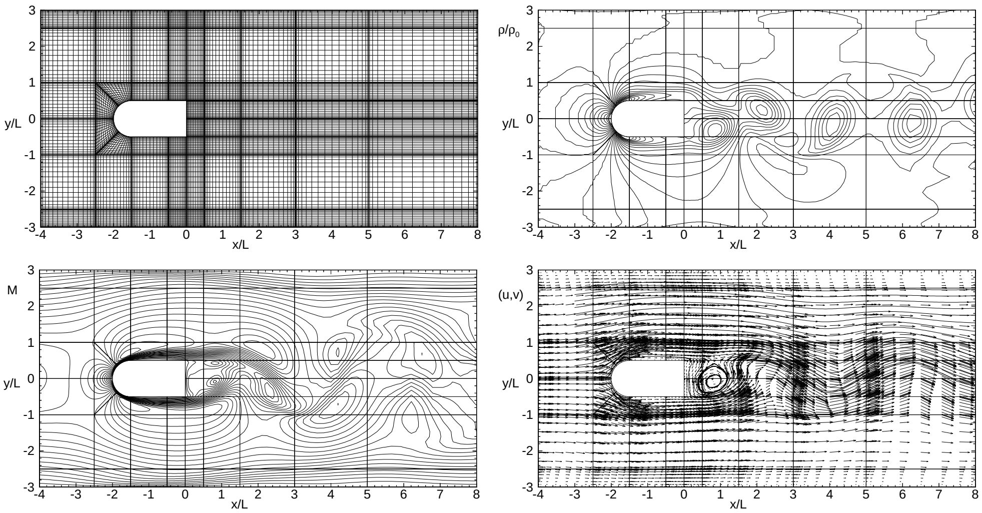
We note that the above result is strictly valid only in the case in which the Jacobian is a constant. However, as we will show shortly, the scheme remains stable also for non-constant Jacobians, thus establishing a stable method for approximating the compressible Navier-Stokes equations in a general quadrilateral domain. A similar result has been established for supersonic inflow and outflow conditions and can be found in [Hes97b].

The result stated in Theorem 14.2 is valid also for the vertices of the quadrilateral. However, at a point where several vertices meet, one has to determine which element is upstream and which is downstream in order to pass the appropriate information between the vertices. For this purpose we define the two vectors, $\boldsymbol{n}_\xi = \xi\nabla\xi$ and $\boldsymbol{n}_\eta = \eta\nabla\eta$. A vertex, say $(\xi, \eta) = (-1, -1)$, can then be identified as the upstream vertex provided $\boldsymbol{u} \cdot \boldsymbol{n}_\xi > 0$ and $\boldsymbol{u} \cdot \boldsymbol{n}_\eta > 0$. In a similar fashion, we may identify the downstream element by reversing the inequalities. The conditions for this test are summarized in the Appendix. Contrary to some previously proposed schemes (see e.g. [Kop91]), this approach handles any number of domains coming together, as the upstream and downstream domains are uniquely identified through the signs of the above scalar-products. For the boundary conditions of the viscous part, we use the average of the Cartesian derivatives across the vertex.

For temporal integration, we use a 3rd-order Runge-Kutta with the boundary conditions being imposed at the intermediate time-steps. Following completion of each time-step, we enforce global continuity and we use the solution at the previous time-step as the solution around which we linearize at the sub-domain boundaries, while the exact solution is used at the open boundaries. The time-step is computed adaptively as

$$\Delta t \leq \mathrm{CFL} \times \min_{\mathbf{x}\in\Omega} \left[|\chi \cdot \boldsymbol{u}| + c\sqrt{\chi \cdot \chi} + \frac{2\gamma}{\mathrm{Pr}\mathrm{Re}_{\mathrm{ref}}} \frac{\mu}{\rho} \chi \cdot \chi \right]^{-1} \quad ,$$

Figure 1 Fragment of the grid used for the flame holder computation. Also shown is the instantaneous density, ρ/ρ_0, the Mach number, M, and the velocity field, $\mathbf{u}$.



where χ is the grid-distortion vector, $\boldsymbol{\chi} = [|\xi_x|/\Delta\xi_i + |\eta_x|/\Delta\eta_j, |\xi_y|/\Delta\xi_i + |\eta_y|/\Delta\eta_j]$ with $\Delta\xi_i$ signifying the local grid size and similarly for $\Delta\eta_j$.

4 Numerical Examples and Remarks

We have implemented the proposed scheme in order to confirm the theoretical results obtained for the linearized, constant coefficient Navier-Stokes equations. In [Hes97a, Hes97b] we presented several solutions of steady state flows, confirming the spectral accuracy of the proposed scheme. However, to emphasize the ability to handle truly unsteady flow, we consider here a problem of some practical importance.

We consider the flow around a flame holder embedded in a narrow channel. This geometry can be viewed as a prototype combustion chamber in a high-speed ram-jet. However, although the engine is designed to perform at supersonic speeds, the flow in the combustion chamber remains purely subsonic. We consider the geometry pictured in Fig. 1, with the base hight of the flame holder being, $L = 12.7$ mm. The flame holder is embedded in a narrow channel with a total height of only $6L$. The full length of the computational domain is $25L$, i.e., Fig. 1 shows only a part of the computational domain.

All walls are assumed to be isothermal, no-slip wall, being held at a stagnation temperature of $T_0 = 300°K$. The free-stream Reynolds number is 250 and the Mach number is 0.4, ensuring that the flow remains subsonic.

The total computation uses 104 elements, each employing a polynomial expansion of order 14. The open boundaries are held at the free-stream values with a laminar, parabolic inflow and outflow velocity profile and the pressure drop computed self-consistently.

Figure 1 clearly illustrates the well known von Karman vortex street rear of the bluff body and the boundary layers at the wall. We also note that all fields are smooth across sub-domain boundaries, including the vertices.

Although these results are of a qualitative nature they confirm the stability of the complete scheme for general curvilinear elements, the validity of the treatment of the vertices and the efficacy of the scheme for the study of unsteady compressible flows in complex geometries.

We have not addressed the question of efficient implementation. However, we recall that the patching of sub-domains and treatment of physical boundaries is purely local in time and space, i.e., the algorithm lends it self to efficient implementation on parallel computers with distributed memory. This will be of significant importance when future attention is directed towards the solution of unsteady three-dimensional problems in complex geometries.

Acknowledgement

The support by DOE grant DE-FG02-95ER25239 and NSF grant ASC-9504002 is greatly appreciated.

Appendix

To ensure stability of the semi-discrete scheme we must choose the parameters, $\boldsymbol{n}$ and $\tilde{\omega}$, appropriately. Moreover, we need to establish the proper conditions for identifying a vertex as an upstream or downstream vertex.

Let us first define the vectors, $\boldsymbol{n}_\xi = \xi\nabla\xi$ and $\boldsymbol{n}_\eta = \eta\nabla\eta$. We will also introduce the two variables, ω_ξ and ω_η. The actual value of these parameters are resolution as well as method dependent.

For *Legendre* methods, we have $\omega_\xi = 2/(N_\xi(N_\xi+1))$ and $\omega_\eta = 2/(N_\eta(N_\eta+1))$, where N_ξ and N_η represents the resolution along ξ and η, respectively.

For *Chebyshev* methods, on the other hand, we have $\omega_\xi = N_\xi^{-2}$ and $\omega_\eta = N_\eta^{-2}$. The appropriate values of the parameter, $\tilde{\omega}$, and the outward pointing normal vector, $\boldsymbol{n}$, required to construct stable schemes along edges and vertices of the quadrilateral is given below.

We also give the condition for determining whether a vertex is indeed upstream. For this purpose, we introduce the convective velocity, $\boldsymbol{u}$. The conditions for naming a purely downstream vertex is obtained by reversing the inequalities.

		τ − Parameters		Outflow Conditions	
ξ	η	$\tilde{\omega}$	$\boldsymbol{n}$	$\boldsymbol{u}\cdot\boldsymbol{n}_\xi$	$\boldsymbol{u}\cdot\boldsymbol{n}_\eta$
±1	$\cdot$	ω_ξ	$\boldsymbol{n}_\xi$	>0	$-$
$\cdot$	±1	ω_η	$\boldsymbol{n}_\eta$	$-$	>0
±1	±1	$\omega_\xi\omega_\eta$	$\omega_\eta\boldsymbol{n}_\xi + \omega_\xi\boldsymbol{n}_\eta$	>0	>0

REFERENCES

[FG88] Funaro D. and Gottlieb D. (1988) A new method of imposing boundary conditions in pseudospectral approximations of hyperbolic equations. *Math. Comp.* 51: 599–613.

[Fun92] Funaro D. (1992) *Polynomial Approximation of Differential Equations*, volume m8 of *Lecture Notes in Physics*. Springer Verlag, Berlin.

[GH73] Gordon W. J. and Hall C. A. (1973) Transfinite element methods: Blending-function interpolation over arbitrary curved element domains. *Numer. Math.* 21: 109–129.

[Han93] Hanley P. (1993) A strategy for the efficient simulation of viscous compressible flows using a multi-domain pseudospectral method. *J. Comp. Phys* 108: 153–158.

[Hes97a] Hesthaven J. S. (1997) A stable penalty method for the compressible Navier-Stokes equations. II. One dimensional domain decomposition schemes. *SIAM J. Sci. Comp.* Accepted.

[Hes97b] Hesthaven J. S. (1997) A stable penalty method for the compressible Navier-Stokes equations. III. Multi dimensional domain decomposition schemes. *SIAM J. Sci. Comp.* Accepted.

[HG96] Hesthaven J. S. and Gottlieb D. (1996) A stable penalty method for the compressible Navier-Stokes equations. I. Open boundary conditions. *SIAM J. Sci. Comp.* 17(3): 579–612.

[KK96] Kopriva D. and Kolias J. H. (1996) A conservative staggered-grid Chebyshev multidomain method for compressible flows. *J. Comp. Phys.* 125(1): 244–261.

[KL89] Kreiss H. O. and Lorenz J. (1989) *Initial-Boundary Value Problems and the Navier-Stokes Equations*, volume 136 of *Pure and Applied Mathematics*. Academic Press, Inc., San Diego.

[Kop91] Kopriva D. (1991) Multidomain spectral solution of the Euler gas-dynamics equations. *J. Comp. Phys* 96: 428–450.

[Kop93] Kopriva D. (1993) A multi-domain spectral method for viscous compressible flows. *AIAA J.* 31: 3376–3384.

[Sch79] Schlichting H. (1979) *Boundary-Layer Theory*. McGraw-Hill Classics Textbook Reissue Series. McGraw-Hill, New York, 7th edition.

[Str77] Strikwerda J. C. (1977) Initial boundary value problems for incompletely parabolic systems. *Comm. Pure Appl. Math.* 30: 797–822.

[WBH75] Warming R. F., Bean R. M., and Heytt B. J. (1975) Diagonalization and simultaneous symmetrization of the gas-dynamics matrices. *Math. Comp.* 29: 1037–1045.

15

Preconditioners for the Boundary Element Method in Three Dimensions

Norbert Heuer

1 Introduction

In this paper we consider preconditioners for linear systems arising from the boundary element method (BEM) for solving partial differential equations in $\mathbf{R}^3$. We report on new results recently obtained, partly in joint work with Ernst P. Stephan.

The boundary element method consists in solving an integral equation formulation of a boundary value problem by the Galerkin method. All the integrals are defined on the boundary of the domain under consideration and, thus, only the boundary of the domain needs to be discretized. Therefore, this method is extremely well suited for transmission problems and exterior boundary value problems where unbounded domains occur.

We deal with first kind integral equations that stem from representation formulae for the solutions of the boundary value problems. Typically there appear hypersingular and weakly singular operators that have to be numerically inverted. They are of orders $+1$ and -1, respectively. In this paper we consider Laplace's equation in $\mathbf{R}^3$. Then the hypersingular operator

$$Du(x) := \frac{1}{4\pi}\frac{\partial}{\partial n_x}\int_\Gamma u(y)\frac{\partial}{\partial n_y}\frac{1}{|x-y|}\,dS_y$$

and the single layer potential operator

$$Vu(x) := \frac{1}{4\pi}\int_\Gamma \frac{u(y)}{|x-y|}\,dS_y$$

are positive definite. To extend our results to more general problems, e.g., to the Helmholtz equation, one has to deal with indefinite integral operators, the main symbols being the operators D and V. Regarding this generalization we refer to [CW92] for the finite element method and to [ST] for the boundary element method in two dimensions.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

For the preconditioning of linear systems arising from the BEM in two dimensions we refer to [Heu96b, TS96, ST95, HST95].

An outline of the paper is as follows. In §2 we introduce the abstract problem under consideration and the boundary element Galerkin method for solving it. Section 3 is dedicated to the hypersingular operator. In §3 we study a general multilevel method to precondition the linear systems arising from the h-version of the BEM. The obtained condition number is almost bounded independently of the number of levels. For a general 2-level method we get a condition number which behaves logarithmically in the ratio H/h. Here H and h are the mesh sizes of the coarse and the fine level meshes, respectively. For the p-version we consider a preconditioner based on an overlapping decomposition and this results in a bounded condition number. For a direct sum decomposition of the ansatz space we use discretely harmonic basis functions and obtain a condition number which is bounded polylogarithmically in the polynomial degree. In §4 we present an almost optimal bound for the condition number in case of the p-version of the BEM for the single layer potential operator. This preconditioner is based on a general decomposition of the ansatz space. Section 5 reports on some numerical results supporting the theoretical estimates.

2 Boundary Element Method

The weak formulation under consideration is the following:
For given $f \in H^{-\alpha/2}(\Gamma)$, find $u \in \tilde{H}^{\alpha/2}(\Gamma)$ such that

$$\langle Au, \phi\rangle_{L^2(\Gamma)} = \langle f, \phi\rangle_{L^2(\Gamma)} \quad \textit{for all } \phi \in \tilde{H}^{\alpha/2}(\Gamma). \tag{1}$$

Here, A is a positive definite operator of order α mapping $\tilde{H}^{\alpha/2}(\Gamma)$ continuously onto $H^{-\alpha/2}(\Gamma)$. In case of the hypersingular operator $\alpha = 1$ and for the weakly singular operator we have $\alpha = -1$. The space $\tilde{H}^{1/2}(\Gamma)$, which is also denoted by $H^{1/2}_{00}(\Gamma)$ in the finite element literature, is the interpolation space half-way between $L^2(\Gamma)$ and $H^1_0(\Gamma)$. The space $H^{-1/2}(\Gamma)$ is the dual space of $\tilde{H}^{1/2}(\Gamma)$ and, vice versa, $\tilde{H}^{-1/2}(\Gamma)$ is the dual space of $H^{1/2}(\Gamma)$ which is the interpolation space half-way between $L^2(\Gamma)$ and $H^1(\Gamma)$. We assume Γ to be a flat rectangular screen in $\mathbf{R}^3$. The extension of our results to arbitrary polyhedral surfaces consisting of rectangular pieces is straight forward.

In the case of $A = D$ eq. (1) models a Neumann problem for the Laplacian in $\mathbb{R}^3 \setminus \bar{\Gamma}$ where the jump across Γ of the normal derivative of the solution is given. When $A = V$ eq. (1) represents the Dirichlet problem where the jump across Γ of the trace of the solution is given.

The Galerkin scheme for solving (1) reads as follows:
For a given N-dimensional subspace X_N of $\tilde{H}^{\alpha/2}(\Gamma)$, find $u_N \in X_N$ such that

$$\langle Au_N, \phi\rangle_{L^2(\Gamma)} = \langle f, \phi\rangle_{L^2(\Gamma)} \quad \textit{for all } \phi \in X_N. \tag{2}$$

To construct X_N we use a uniform mesh Γ_h on Γ of rectangles of size h.

First let us consider the case $\alpha = 1$. For the h-version of the boundary element method we use piecewise bilinear functions which have the value one at one interior node and vanish at the remaining nodes of Γ_h. Note that the condition $X_N \subset \tilde{H}^{1/2}(\Gamma)$ requires continuous functions which are zero on the boundary of Γ. For the p-version of the boundary element method we use piecewise polynomials of degree p on the mesh Γ_h. As basis functions we take affine images of all combinations of tensor products of piecewise linear functions and of antiderivatives of Legendre polynomials. We note that, to our best knowledge, regarding the efficiency of the implementation there are no algorithms making special use of the Lagrangian interpolation polynomials in the Legendre-Gauss-Lobatto nodes. These functions can be efficiently used in the spectral method which is a special p-version of the finite element method. The efficiency of the spectral method heavily relies on the fact that differential operators have to be discretized. Therefore, in view of approximation properties and the efficiency of the implementation, it is opportune to use the antiderivatives of the Legendre polynomials to construct basis functions for degrees larger than 1.

For the weakly singular operator, i.e. $\alpha = -1$, we just consider the p-version. Then our ansatz spaces X_N are constructed by using affine images of tensor products of Legendre polynomials up to degree p on a mesh Γ_h. We note that $X_N \subset \tilde{H}^{-1/2}(\Gamma)$ does not require continuous functions.

To refer to the parameters h and p of our ansatz space we use the notations

$$X_N = S_p^1(\Gamma_h) \subset \tilde{H}^{1/2}(\Gamma)$$

and

$$X_N = S_p^0(\Gamma_h) \subset \tilde{H}^{-1/2}(\Gamma).$$

In either case, $\alpha = 1$ and $\alpha = -1$, the stiffness matrix in (2), which is also denoted by A, is positive definite since both operators D and V are positive definite. Therefore, the Galerkin method converges quasi-optimally in the energy norm and the method of choice to solve the linear system (2) is the conjugate gradient algorithm. In order to reduce the numbers of iterations which are necessary to solve (2) up to a given accuracy we use preconditioners. By referring to the additive Schwarz frame work, they are defined via decompositions of X_N. To be precise we define the additive Schwarz operator P for a decomposition $X_N = X_1 \cup X_2 \cup \cdots \cup X_k$ by the sum of the projections $P_i : X_N \to X_i$, $P = \sum_{i=1}^k P_i$. All the projections are performed with respect to the bilinear form $\langle A\cdot, \cdot\rangle_{L^2(\Gamma)}$. That means we use exact solvers for all the subspaces of X_N. For practical problems they can be replaced by inexact solvers. The additive Schwarz operator P represents the preconditioned stiffness matrix of the linear system and the aim is to find decompositions of X_N which result in small conditions numbers of P. For a survey on additive Schwarz methods we refer to [CM94].

3 Preconditioners for the Hypersingular Operator

h-version

As mentioned above we have to deal with the ansatz space $X_N = S_p^1(\Gamma_h)$ of continuous functions. For the h-version we take the polynomial degree $p = 1$. To define the multilevel preconditioner we consider L mesh sizes $h_1, h_2, \ldots, h_L$ with $h_{l-1} = 2h_l$, $l = 2, \ldots, L$, and $h_L = h$. In a first step we decompose $S_1^1(\Gamma_h)$ into L levels,

$$S_1^1(\Gamma_h) = S_1^1(\Gamma_{h_1}) \cup S_1^1(\Gamma_{h_2}) \cup \cdots \cup S_1^1(\Gamma_{h_L}).$$

This is an overlapping decomposition since, for $1 \leq l \leq m \leq L$, we have $S_1^1(\Gamma_{h_l}) \subset S_1^1(\Gamma_{h_m})$ due to the relations $h_{l-1} = 2h_l$, $l = 2, \dots, L$. In a second step we totally decompose the subspaces of the different levels except of the coarsest subspace,

$$S_1^1(\Gamma_{h_l}) = S_{1,1}^1(\Gamma_{h_l}) \cup S_{1,2}^1(\Gamma_{h_l}) \cup \cdots \cup S_{1,N_{h_l}}^1(\Gamma_{h_l}).$$

Here each subspace $S_{1,i}^1(\Gamma_{h_l})$ is spanned by exactly one piecewise bilinear basis function on the mesh Γ_{h_l} and N_{h_l} is the dimension of $S_1^1(\Gamma_{h_l})$. The final multilevel decomposition looks like

$$S_1^1(\Gamma_h) = S_1^1(\Gamma_{h_1}) \cup \cup_{l=2}^L \left(S_{1,1}^1(\Gamma_{h_l}) \cup \cdots \cup S_{1,N_{h_l}}^1(\Gamma_{h_l})\right). \tag{3}$$

This means that we use an exact solver for the whole subspace on the coarsest level and that we just use the diagonal preconditioner on all the finer levels. In the 2-level case the coarsest subspace is relatively large and using more levels this subspace becomes smaller. However, in the latter case, the amount of overlapping in the overall decomposition increases. From [Heua] we cite the following result whose proof is a generalization of the theory in [DW91, Zha92, TS96].

Theorem 1 *The additive Schwarz operator corresponding to* (3) *has a condition number which is bounded by*

$$\kappa(P) \leq Ch^{-\epsilon}.$$

The constant C is independent of h, the mesh size of the finest level, and of the number of levels L.

We note that the term $h^{-\epsilon}$ in the estimate of the condition number is due to the singularities of the exact solution of our problem at the boundary of the screen Γ. In the case of a closed surface the solution is more regular and the term $h^{-\epsilon}$ does not appear.

To get rid of both, the use of a large coarse subspace and a huge overlapping, we also consider general 2-level methods where one has a coarse mesh Γ_H which is almost independent of the fine mesh Γ_h, the only restriction being the compatibility. The used decomposition is given by

$$S_1^1(\Gamma_h) = S_1^1(\Gamma_H) \cup S_1^1(\Gamma_H \cap \Gamma_h) \cup \cup_{j=1}^{J_H} S_1^1(\Gamma_h \cap \Gamma_j). \tag{4}$$

The space $S_1^1(\Gamma_H)$ consists of the usual continuous piecewise bilinear functions on the mesh Γ_H of size H. $S_1^1(\Gamma_H \cap \Gamma_h)$ is the so-called wirebasket space which is spanned by the piecewise bilinear hat functions of $S_1^1(\Gamma_h)$ which are associated with the nodes of the fine mesh which are on the grid of the coarse mesh. The spaces $S_1^1(\Gamma_h \cap \Gamma_j)$ are spanned by the piecewise bilinear hat functions which are associated with the nodes interior to the restricted meshes $\Gamma_h|_{\Gamma_j}$, $j = 1, \dots, J_H$. Here, Γ_j, $j = 1, \dots, J_H$, are the elements of the coarse mesh Γ_H. The result is the following, see [HS].

Theorem 2 *The condition number of the additive Schwarz operator P which is defined by the decomposition* (4) *is bounded by*

$$\kappa(P) \leq C(1 + \log \frac{H}{h})$$

where the constant $C > 0$ is independent of the coarse and fine mesh sizes H and h.

Thus, for this preconditioner, we have bounded condition numbers if the ratio H/h is fixed.

p-version

We consider a fixed rectangular mesh Γ_h and take affine images of tensor products of piecewise linear functions and of antiderivatives of Legendre polynomials as basis functions. For the following overlapping decomposition, which has been investigated for the finite element method in [Pav94], we obtain bounded condition numbers of the corresponding additive Schwarz operator:

$$S_p^1(\Gamma_h) = S_1^1(\Gamma_h) \cup S_p^1(\Gamma_h \cap \Gamma_1') \cup \cdots \cup S_p^1(\Gamma_h \cap \Gamma_{N_h}'). \tag{5}$$

The so-called coarse grid space $S_1^1(\Gamma_h)$ is just the space of the h-version. The remaining spaces are subspaces localized at the neighborhoods of the interior nodes. More precisely $S_p^1(\Gamma_h \cap \Gamma_j')$ is the space of piecewise polynomials of degree p which are globally continuous and which have support contained in the elements adjacent to the node with number j. Therefore, subspaces for adjacent nodes may have common functions and in that case the corresponding blocks of the stiffness matrix overlap.

Theorem 3 [Heua] *The condition number of the additive Schwarz operator P which is defined by the decomposition* (5) *is bounded.*

Since the subspaces $S_p^1(\Gamma_h \cap \Gamma_j')$ are rather large for large polynomial degree p one is interested in further splitting the corresponding blocks. Due to the tensor product structure of the basis functions one has a natural decomposition into subspaces of functions which are associated with nodes, edges and elements, separately. However, it is well known that one cannot take the usual nodal hat functions for such a splitting in higher dimensions. This would result in large condition numbers, cf. [BCMP91]. Therefore, in order to use a nonoverlapping decomposition, one has to consider well behaved basis functions, i.e., functions with small energy. As nodal basis functions we take tensor products of the polynomial of degree p which is defined by

$$\|\varphi_0\|_{L^2(-1,1)} = \min_{\varphi \text{ has degree } p} \|\varphi\|_{L^2(-1,1)}, \quad \varphi_0(1) = 1,\ \varphi_0(-1) = 0.$$

The basis functions related to the edges and to the interior of the elements are defined as discrete tensor product solutions in the weak sense of the Laplace equation. For details we refer to [PW96] and [Heub]. The decomposition is as follows:

$$S_p^1(\Gamma_h) = X_0 \oplus X_1 \oplus \cdots \oplus X_{J_h}. \tag{6}$$

Here $X_j = S_p^1(\Gamma_h) \cap \tilde{H}^{1/2}(\Gamma_j)$, $j = 1, \ldots, J_h$, where Γ_j is an element of the mesh Γ_h. X_0 is the global space of the remaining functions which are associated with the nodes and the edges of the mesh. This space is called the wirebasket space.

Theorem 4 [Heub] *The condition number of the additive Schwarz operator P defined by the decomposition* (6) *is bounded by*

$$\kappa(P_W) \le C(1 + \log p)^2.$$

The constant C is independent of the mesh size h and the polynomial degree p.

As shown in [Heub] a similar result holds even for a modified diagonal preconditioner which includes a small block of global functions. Of course, here we also have to use the special discretely harmonic basis functions.

4 Preconditioner for the Weakly Singular Operator

We only study the p-version of the BEM for the single layer potential operator. We use quasi-uniform rectangular meshes of size h on Γ and take discontinuous piecewise polynomials of degree p for the boundary element space $X_N = S_p^0(\Gamma_h)$. We decompose

$$S_p^0(\Gamma_h) = S_p^0(\Gamma_1) \cup \cdots \cup S_p^0(\Gamma_J). \tag{7}$$

The space $S_p^0(\Gamma_j)$ is the restriction of $S_p^0(\Gamma_h)$ onto a subdomain Γ_j where $\bar{\Gamma} = \cup_{j=1}^J \bar{\Gamma}_j$ is a, possibly overlapping, decomposition of Γ. From [Heu96a] we cite the following result.

Theorem 5 *For any $\epsilon > 0$ there exists a constant $C > 0$ such that the condition number of the additive Schwarz operator defined by the decomposition* (7) *is bounded by*

$$\kappa(P) \leq Cp^\epsilon.$$

5 Numerical Results

In this section we present some numerical experiments for the preconditioners defined in the previous sections. We choose the domain Γ to be the square plate $(-1/2, 1/2)^2 \times \{0\}$. For the p-version we use a uniform mesh of 9 elements. Tables 1 and 2 collect some results for the hypersingular operator. Table 1 lists the condition numbers and extremum eigenvalues of the 2-level method for the h-version. As predicted by Theorem 1 the condition numbers are almost bounded. Table 2 shows the results for the p-version. As stated by Theorem 3 the overlapping decomposition produces bounded condition numbers. The numbers for the nonoverlapping decomposition which belongs to the wirebasket preconditioner are just slightly increasing as predicted by Theorem 4. Finally, Table 3 shows the results for the p-version with the single layer potential operator. They are covered by the statement of Theorem 5 and, for the overlapping decomposition, the condition numbers even appear to be bounded. For the nonoverlapping decomposition we simply used the elements as subdomains and for the overlapping decomposition we used patches of 4 elements as subdomains.

REFERENCES

[BCMP91] Babuška I., Craig A., Mandel J., and Pitkäranta J. (1991) Efficient preconditioning for the p-version finite element method in two dimensions. *SIAM J. Numer. Anal.* 28(3): 624–661.

		2-level ASM		
$1/h$	N	κ	$\lambda_{\min}$	$\lambda_{\max}$
4	9	2.58	0.74	1.92
6	25	3.25	0.65	2.10
8	49	3.51	0.61	2.14
10	81	3.62	0.59	2.14
12	121	3.70	0.58	2.14
14	169	3.75	0.58	2.15
16	225	3.78	0.57	2.16

Table 1 h-version of the BEM with the hypersingular operator. Condition numbers and eigenvalues for the 2-level preconditioner.

		overlapping dec.			nonoverlapping dec.		
p	N	κ	$l_{\min}$	$l_{\max}$	κ	$l_{\min}$	$l_{\max}$
1	4	1.12	1.86	2.08	1.00	1.00	1.00
2	25	4.96	0.83	4.10	5.01	0.32	1.61
3	64	4.73	0.87	4.10	6.21	0.26	1.64
4	121	4.64	0.89	4.13	8.06	0.21	1.70
5	196	4.54	0.91	4.13	8.66	0.20	1.71
6	289	4.49	0.92	4.13	9.91	0.18	1.74

Table 2 p-version of the BEM with the hypersingular operator. Condition numbers and eigenvalues for the overlapping and the nonoverlapping decompositions.

[CM94] Chan T. F. and Mathew T. P. (1994) Domain decomposition algorithms. *Acta Numerica* pages 61–143.

[CW92] Cai X.-C. and Widlund O. B. (1992) Domain decomposition algorithms for indefinite elliptic problems. *SIAM J. Sci. Stat. Comput.* 13: 243–258.

[DW91] Dryja M. and Widlund O. B. (1991) Multilevel additive methods for elliptic finite element problems. In Hackbusch W. (ed) *Parallel Algorithms for Partial Differential Equations (Proc. of the Sixth GAMM-Seminar, Kiel, Germany, January 19–21, 1990)*, pages 58–69. Vieweg, Braunschweig, Germany.

[Heua] Heuer N.Additive Schwarz methods for hypersingular integral equations in $\mathbb{R}^3$. Submitted for publication.

[Heub] Heuer N.An iterative substructuring method for the p-version of the boundary element method for hypersingular integral equations in three dimensions. Submitted for publication.

[Heu96a] Heuer N. (1996) Additive Schwarz methods for weakly singular integral equations in $\mathbb{R}^3$ – the p-version. In Hackbusch W. and Wittum G. (eds) *Boundary Elements: Implementation and Analysis of Advanced Algorithms*, volume 54 of *Notes on Numerical Fluid Mechanics*, pages 126–135. Vieweg-Verlag, Braunschweig, Wiesbaden. Proceedings of the 12th GAMM-Seminar, Kiel, January 1996.

[Heu96b] Heuer N. (1996) Efficient algorithms for the p-version of the boundary

		overlapping dec.			nonoverlapping dec.		
p	N	κ	$l_{\min}$	$l_{\max}$	κ	$l_{\min}$	$l_{\max}$
0	9	4.08	0.98	4.00	7.02	0.43	3.02
1	27	4.35	0.92	4.00	8.20	0.38	3.14
2	54	4.21	0.95	4.00	11.40	0.30	3.36
3	90	4.26	0.94	4.00	12.22	0.28	3.39
4	135	4.22	0.95	4.00	14.60	0.24	3.46
5	189	4.24	0.94	4.00	15.26	0.23	3.47
6	252	4.22	0.95	4.00	17.14	0.20	3.50

Table 3 p-version of the BEM with the weakly singular operator. Condition numbers and eigenvalues for overlapping and nonoverlapping decompositions.

element method. *J. Integral Equations Appl.* 8(3). To appear.

[HS] Heuer N. and Stephan E. P.Iterative substructuring for hypersingular integral equations in $\mathbb{R}^3$. Submitted for publication.

[HST95] Heuer N., Stephan E. P., and Tran T. (1995) Multilevel additive Schwarz method for the p- and hp-version boundary element method. Technical Report AMR95/37, The University of New South Wales.

[Pav94] Pavarino L. F. (1994) Additive Schwarz methods for the p-version finite element method. *Numer. Math.* 66: 493–515.

[PW96] Pavarino L. F. and Widlund O. B. (1996) A polylogarithmic bound for an iterative substructuring method for spectral elements in three dimensions. *SIAM J. Numer. Anal.* 33(4): 1303–1335.

[ST] Stephan E. P. and Tran T.Domain decomposition algorithms for indefinite hypersingular integral equations – the h- and p-versions. *SIAM J. Sci. Stat. Comput.* To appear.

[ST95] Stephan E. P. and Tran T. (1995) Additive Schwarz method for the p-version boundary element method. Technical Report AMR95/13, The University of New South Wales.

[TS96] Tran T. and Stephan E. P. (1996) Additive Schwarz method for the h-version boundary element method. *Appl. Anal.* 60: 63–84.

[Zha92] Zhang X. (1992) Multilevel Schwarz methods. *Numer. Math.* 63: 521–539.

16

Combining Waveform Relaxation and Domain Decomposition

Sigitas Keras

1 Introduction

Several techniques with inherent parallelism are available for the solution of parabolic equations, and among the most successful are *Domain Decomposition* (DD) and *Waveform Relaxation* (WR) methods. The main goal of this paper is to demonstrate that it is possible to combine these techniques into a single algorithm, thus reducing the computational complexity and the time required to obtain the computational solution of parabolic equations.

Waveform relaxation was first suggested in the late 19th century by Picard and Lindelöf ([Pic93, Lin94]) and has been subjected to much recent interest as a practical method for the solution of stiff ODEs after the publication of the paper by Lelarasmee and coworkers [LRSV82]. Recent work in this field includes papers by Nevanlinna, Zennaro, Bjørhus and others (see [MN87a, Bjø95, BZ93, Lum92]). It can be described as an iterative method to solve an initial value problem

$$\frac{du}{dt} = f(u), \quad y(0) = y_0.$$

At each step we compute the solution of the equation

$$\frac{du}{dt}^{(n+1)} = \tilde{f}(u^{(n+1)}, u^{(n)}), \quad u^{(n+1)}(0) = y_0, \tag{1}$$

where the function $\tilde{f}$ satisfies the identity

$$\tilde{f}(v, v) = f(v).$$

In this paper we only consider the case where f is a linear operator, which throughout will be denoted by $f(v) = -Av$, and $\tilde{f}(v, w) = -Pv + Qw$, where P and Q are linear operators and $A = P - Q$. In this case (1) lends itself to the following form:

$$\frac{du}{dt}^{(n+1)} + Pu^{(n+1)} = Qu^{(n)} + f, \quad u^{(n+1)}(0) = y_0. \tag{2}$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Solving (2) explicitly by integration of constants, $u^{(n+1)}$ can be formally written as

$$u^{(n+1)} = \mathcal{K}u^{(n)} + \phi,$$

where

$$\mathcal{K}u(t) = \int_0^t e^{(s-t)P} Qu(s)ds,$$
$$\phi(t) = e^{-tP}u_0 + \int_0^t e^{(s-t)P} f(s)ds,$$

and it follows from the Banach fixed point theorem that the method (2) converges for all f and u^0 if and only if $\rho(\mathcal{K}) < 1$, where ρ denotes the spectral radius of the operator.

An extensive theory for this iterative procedure and its discrete version has been developed in [MN87a] and [MN87b]. In particular, necessary and sufficient conditions for the convergence of this method have been established. The following result has been proved in [MN87b].

Theorem 1. Suppose that all the eigenvalues of A and P have positive real parts. Then the spectral radius of $\mathcal{K}$ can be represented by means of formula

$$\rho(\mathcal{K}) = \max_{\xi \in R} \rho((i\xi I + P)^{-1}Q). \tag{3}$$

This result can be applied to semidiscretized linear parabolic PDEs. The following theorem has been proved in [Ker95b].

Theorem 2. Consider the diffusion equation

$$u_t - \nabla(a(x)\nabla u(x)) = f, \quad (x,t) \in \Omega \times (0,\infty), \tag{4}$$
$$u(0,x) = u_0(x), \quad x \in \Omega, \tag{5}$$
$$u(t,x) = 0, \quad (x,t) \in \partial\Omega \times [0,\infty). \tag{6}$$

where Ω is a rectangular domain in $\mathbb{R}^d$. Let A be a discretization of the elliptic operator $-\nabla(a(x)\nabla)$, $0 < a_- \leq a(x) \leq a_+ < \infty$ and P be a discretization of the operator $-\nabla(b(x)\nabla)$ for some function b such that $0 < b_- \leq b(x) \leq b_+ < \infty$ and let both A and P satisfy the following assumptions:

1. A and P are positive definite
2. $c\langle Pu, u\rangle < \langle Au, u\rangle < C\langle Pu, u\rangle$ for any vector $u \neq 0$, provided that $cb(x) < a(x) < Cb(x)$ for all $x \in \Omega$, where $c, C \in \mathbb{R}$

Then the method (2) converges if $\max\left|\frac{a(x)-b(x)}{b(x)}\right| < 1$ and $\rho(\mathcal{K}) < \max\left|\frac{a(x)-b(x)}{b(x)}\right|$.

In particular, when $b(x)$ is a constant, one can deduce that

$$\rho(\mathcal{K}) \leq \max\left|\frac{a(x)-C}{C}\right| \tag{7}$$

and

$$\min_{C \in \mathbb{R}^+} \rho(\mathcal{K}) \leq \frac{a_+ - a_-}{a_+ + a_-}. \tag{8}$$

In rectangular domains this choice of $b(x)$ makes the matrix P Toeplitz or block Toeplitz, which allows a fast solution of the subproblem (2). One may ask what other choices can be made for $b(x)$. In this paper we consider a natural generalization in which $b(x)$ is a piecewise constant function. In order to exploit the structure of the matrix P, we propose to solve the WR equations using domain decomposition techniques.

2 Nonoverlapping Domain Decomposition

Consider the equation

$$Au = f, \tag{9}$$

where u and f are vectors defined on a grid in the domain Ω. We assume that Ω is divided into two subdomains Ω_1 and Ω_2 separated by the boundary B. We subdivide vectors u and f

$$u = \begin{pmatrix} u_1 \\ u_2 \\ u_B \end{pmatrix}, \quad f = \begin{pmatrix} f_1 \\ f_2 \\ f_B \end{pmatrix}, \tag{10}$$

where indices $1, 2$ and B denote restrictions of the vectors to the domains Ω_1, Ω_2 and the boundary B respectively. Similarly, we can write

$$A = \begin{pmatrix} A_{11} & A_{12} & A_{1B} \\ A_{21} & A_{22} & A_{2B} \\ A_{B1} & A_{B2} & A_{BB} \end{pmatrix},$$

where submatrices A_{ij} satisfy the relationship $\sum_j A_{ij}u_j = f_i$. We assume that matrix A is symmetric, $A = A^T$, and also that $A_{12} = A_{21}^T = O$, which means that there is no interaction between subdomains Ω_1 and Ω_2 other than through the boundary. These assumptions are satisfied by matrices arising from the discretization of elliptic operators which are considered in the present paper. In this case (9) can be written as a set of three independent equations

$$\begin{aligned} Su_B &= \tilde{f}_B, & (11) \\ A_{11}u_1 &= \tilde{f}_1, & (12) \\ A_{22}u_2 &= \tilde{f}_2, & (13) \end{aligned}$$

where

$$\begin{aligned} S &= A_{BB} - A_{B1}A_{11}^{-1}A_{1B} - A_{B2}A_{22}^{-1}A_{2B}, & (14) \\ \tilde{f}_B &= f_B - A_{B1}A_{11}^{-1}u_1 - A_{B2}A_{22}^{-1}u_2, & (15) \\ \tilde{f}_1 &= f_1 - A_{1B}u_B, & (16) \\ \tilde{f}_2 &= f_2 - A_{2B}u_B. & (17) \end{aligned}$$

Equation (11) is solved first, after which we can determine vectors $\tilde{f}_1$ and $\tilde{f}_2$ and solve (12) and (13). Note that (12) and (13) are independent of each other and can be solved simultaneously.

An important part of the algorithm is to be able to solve the equation for the vector u_B efficiently. The matrix S is typically dense and expensive to calculate explicitly. In practice the equation for u_B is solved using the preconditioned conjugate gradient method (PCG), and the rate of convergence depends on the condition number of the preconditioned matrix S. We present here two examples of preconditioners for two-dimensional elliptic problems, for a more elaborate discussion refer to the review paper of Chan and Matthew [CM94].

In order to introduce the preconditioners, we make further assumptions about the structure of the grid on the boundary, which is denoted with a subscript B. Assume that the domain Ω is divided into rectangular subdomains so that the boundary B consists of k edges E_i, $i = 1, \ldots, k$ and vertices V, and the matrix S can be decomposed as follows,

$$S = \begin{pmatrix} S_{E_1E_1} & \ldots & S_{E_1E_k} & S_{E_1V} \\ \vdots & \ddots & \vdots & \vdots \\ S_{E_kE_1} & \ldots & S_{E_kE_k} & S_{E_kV} \\ S_{VE_1} & \ldots & S_{VE_k} & S_{VV} \end{pmatrix}.$$

The first preconditioner is a *block Jacobi* preconditioner

$$M_1 = \begin{pmatrix} S_{E_1E_1} & 0 & \ldots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & S_{E_kE_k} & 0 \\ 0 & \ldots & 0 & S_{VV} \end{pmatrix}.$$

One may expect that the condition number of the preconditioned system, $M_1^{-1}S$, is dependent on the discretization size h as well as on the size of the subdomains H. This is in fact the case as shown in [BPS86],

Theorem 3. There exists a constant C independent of H and h, such that

$$\mathrm{cond}(M_1^{-1}S) \leq CH^{-2}(1 + \ln(H/h)).$$

The other preconditioner which was also introduced in [BPS86] can be written in the form

$$M_2 = \begin{pmatrix} S_{E_1E_1} & 0 & \ldots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & S_{E_kE_k} & 0 \\ 0 & \ldots & 0 & A_V \end{pmatrix}.$$

Here A_V is a matrix resulting from the discretization of the problem only on vertices V. Global coupling between the vertices, introduced by the matrix A_V, substantially reduces the condition number of the matrix S. The following result has been proved in [BPS86].

Theorem 4. There exists a constant C independent of H and h such that

$$\mathrm{cond}(M_2^{-1}S) \leq C(1 + \ln^2(H/h)).$$

3 Domain Decomposition for WR Equations

Consider the waveform relaxation method with a splitting as described in Theorem 2, where the domain Ω is divided into subdomains Ω_i and function $b(x)$ is constant in each of the subdomains Ω_i. Then it is easy to show that

$$\rho(\mathcal{K}) < \max_i \frac{a^i_{\max} - a^i_{\min}}{a^i_{\max} + a^i_{\min}}, \tag{18}$$

where

$$a^i_{\max} = \max_{x \in \Omega_i} a(x), \qquad a^i_{\min} = \min_{x \in \Omega_i} a(x).$$

Indeed, we choose

$$b(x) = \frac{a^i_{\max} + a^i_{\min}}{2}, \quad x \in \Omega_i; \tag{19}$$

then the estimate (18) follows from Theorem 2.

In this way, we obtain a WR method with an improved radius of convergence. There is, however, a price to pay. The main advantage of the waveform relaxation method is due to the fact that the matrix P is easily invertible, so that (2) is easy to solve. For instance, in case of a constant function $b(x)$ we have obtained a block Toeplitz matrix P which can be inverted very fast using FFT techniques. In case of a piecewise constant function $b(x)$ this structure is destroyed. To overcome this difficulty we propose to employ the domain decomposition method. There are several reasons why this can be a promising approach. Firstly, as we have already noted in the previous section, the main computational cost of domain decomposition consists of solving the equation in each subdomain. This can be implemented in parallel. Since $b(x)$ is constant in each subdomain, this results in solving equations with block Toeplitz matrices. Secondly, we employ the domain decomposition method in order to solve the equation for $(n+1)$st iteration of the WR method $u^{(n+1)}$. Since this is not the solution we are seeking but only an iteration, we need not to solve the domain decomposition equations exactly. In other words, we propose to use an inner-outer iteration scheme where inner iteration is performed using DD algorithm and outer iteration is performed using WR method.

We start by stating a result about the convergence of iterative methods for time dependent iterative schemes. Consider a linear system of ordinary differential equations

$$\frac{du}{dt} + Pu = f(t), \qquad u(0) = u_0,$$

where P is a symmetric positive definite matrix. We solve it on a finite time interval $[0, t^*]$ with the θ method

$$\frac{u_{n+1} - u_n}{\Delta t} + \theta P u_{n+1} + (1-\theta) P u_n = f_n, \qquad \theta \geq 1/2,$$

At each time step the resulting linear equation

$$u_{n+1} = \left(\frac{1}{\Delta t} I + \theta P\right)^{-1} \left[\left(\frac{1}{\Delta t} I - (1-\theta) P\right) u_n + f_n\right] \tag{20}$$

Table 1 The performance of the standard parallel solver for the test problem with $\alpha = 0.5$ on large grids.

Grid size	Number of nodes	Time (sec)
$64 \times 64 \times 50$	1	76.22
$64 \times 64 \times 50$	2	41.61
$64 \times 64 \times 50$	4	24.45
$64 \times 64 \times 50$	8	15.63
$64 \times 64 \times 50$	16	12.08
$64 \times 64 \times 50$	32	11.44
$128 \times 128 \times 50$	1	584.1
$128 \times 128 \times 50$	2	300.1
$128 \times 128 \times 50$	4	158.64
$128 \times 128 \times 50$	8	88.83
$128 \times 128 \times 50$	16	54.25
$128 \times 128 \times 50$	32	39.83
$128 \times 128 \times 50$	64	31.58

is solved using an iterative method with a linear rate of convergence ρ,

$$||u_n^{k+1} - u_n|| \leq \rho||u_n^k - u_n||, \;\; k = 1, 2, \ldots,$$

where the superscript k denotes the iteration number. We construct an approximate solution by applying m iterations at each time step and using the new value $v_n = u_{n+1}^m$ in the right hand side of (20). Then the following theorem holds.

Theorem 5. If m is large enough, then the error at time step n, $e_n = v_n - u_n$, satisfies the inequality

$$||e_n|| < C\rho^m$$

where the constant C is independent of m and ρ.

The above theorem can be applied to our problem of combining the waveform relaxation and domain decomposition methods together. Consider the equation

$$\frac{du}{dt} + Au = f,$$

which is solved using the waveform relaxation method (2) as described in Theorem 2, and at each iteration the resulting equation is solved using m iterations of the domain decomposition method. Let Ω be divided into subdomains Ω_i and let $b(x)$ be defined as in (19).

Theorem 6. If m is large enough, then the combined waveform relaxation – domain decomposition method converges.

The proofs of Theorems 5 and 6 are rather technical and are given in full detail in [Ker95a].

Table 2 The performance of the waveform relaxation with Toeplitz splitting for the test problem with $\alpha = 0.5$ on large grids.

Grid size	Number of nodes	Number of iterations	Time (sec)
$64 \times 64 \times 50$	2	10	40.92
$64 \times 64 \times 50$	4	10	26.74
$64 \times 64 \times 50$	8	10	20.35
$64 \times 64 \times 50$	16	10	12.44
$64 \times 64 \times 50$	32	10	8.09
$128 \times 128 \times 50$	2	10	171.8
$128 \times 128 \times 50$	4	10	111.5
$128 \times 128 \times 50$	8	10	83.46
$128 \times 128 \times 50$	16	10	49.44
$128 \times 128 \times 50$	32	10	27.43

4 A Numerical Example

A new numerical method can only be justified if it performs comparably to or better than existing methods. In this section we present a numerical example. All the calculations were carried out on an Intel Paragon computer. Our test problem is a parabolic equation with variable coefficients,

$$\begin{aligned} \frac{\partial u}{\partial t} &= \frac{\partial}{\partial x}\left((1+\alpha \sin 4\pi x \sin 4\pi y)\frac{\partial u}{\partial x}\right) \\ &+ \frac{\partial}{\partial y}\left((1+\alpha \sin 4\pi x \sin 4\pi y)\frac{\partial u}{\partial y}\right), \qquad (21) \\ & \quad (x,y,t) \in \Omega \times (0,1), \quad \Omega = (0,1)\times(0,1) \\ & \quad u(x,y,t) = 0, \quad (x,y,t) \in \partial\Omega \times (0,1), \qquad (22) \\ & \quad u(x,y,0) = \sin \pi x \sin \pi y. \qquad (23) \end{aligned}$$

We solve the above equation using three methods. The first method is a standard Crank–Nicolson scheme. Since this is an implicit scheme, the parallel solver of sparse linear systems is used. The second method is the WR method with Toeplitz splitting as described in [Ker95b]. In this case the matrix P in (2) is block Toeplitz so that parallel FFT solvers are used. Finally, the third method is the combined waveform relaxation domain decomposition method. The parallelization is done by assigning subdomain problems to different processors as well as performing WR iterations on different processors. The preconditioner described in Theorem 3 is used for the solution of resulting Schur problems. While it is not asymptotically optimal, it is easy to implement and provided a good estimate of the method. A better convergence is expected if a better preconditioner is used.

The results of the computations are presented in Tables 1 through 3. They suggest

Table 3 The performance of the combined waveform relaxation domain decomposition method (4 subdomains) for the test problem with $\alpha = 0.5$ on large grids.

Grid size	Number of nodes	Number of iterations	Time (sec)
$64 \times 64 \times 50$	2	8	55.91
$64 \times 64 \times 50$	4	8	35.87
$64 \times 64 \times 50$	8	8	28.63
$64 \times 64 \times 50$	16	8	23.71
$64 \times 64 \times 50$	32	8	18.34
$128 \times 128 \times 50$	2	8	202.1
$128 \times 128 \times 50$	4	8	134.1
$128 \times 128 \times 50$	8	8	91.23
$128 \times 128 \times 50$	16	8	60.23
$128 \times 128 \times 50$	32	8	38.1

that both WR and WRDD methods have better asymptotic properties compared to the standard scheme when the grid size of the space domain increases. In particular, for the grids of the given size, the WR method outperforms the standard solver and the WRDD method performs comparably to it. The number of the iterations required for the convergence is independent on the grid size and the WRDD method requires less iterations than the WR method. The other important feature is that DD can be used as an inner iteration method, so that only several iterations of the DD are needed. In our numerical example we have performed only 3 DD iterations which was sufficient for the convergence of the method.

Acknowledgement

Numerical experiments described in this paper were carried out on Intel Paragon computer at the California Institute of Technology. The author also acknowledges financial support he has received from Magdalene College, Cambridge and EPSRC.

REFERENCES

[Bjø95] Bjørhus M. (1995) A note on the convergence of discretized dynamic iteration. *BIT* 35: 291–296.

[BPS86] Bramble J. H., Pasciak J. E., and Shatz A. H. (1986) An iterative method for elliptic problems on regions partitioned into substructures. *Math. Comp.* 46: 361–369.

[BZ93] Bellen A. and Zennaro M. (1993) The use of Runge-Kutta formulae in waveform relaxation methods. *Appl. Numer. Math.* 11: 95–114.

[CM94] Chan T. F. and Mathew T. P. (1994) Domain decomposition algorithms. *Acta Numerica* 3: 61–143.

[Ker95a] Keras S. (1995) Combining domain decomposition and waveform relaxation. Technical Report NA6, DAMTP, University of Cambridge.

[Ker95b] Keras S. (1995) Waveform relaxation method with Toeplitz operator splitting. Technical Report NA4, DAMTP, University of Cambridge. Submitted to SIAM J. Numer. Anal.

[Lin94] Lindelöf E. (1894) Sur l'application des méthodes d'approximations successives á l'etude des intégrales réeles des équations différentielles ordinaires. *Journal de Mathématiques Pures et Appliquées* 10: 117–128.

[LRSV82] Lelarasmee E., Ruehli A. E., and Sangiovanni-Vincentelli A. L. (1982) The waveform relaxation method for time-domain analysis of large scale integrated circuits. *IEEE Trans. on CAD of IC and Syst.* 1: 131–145.

[Lum92] Lumsdaine A. (1992) *Theoretical and Practical Aspects of Parallel Numerical Algorithms for Initial Value Problems, with Applications.* PhD thesis, Massachusetts Institute of Technology.

[MN87a] Miekkala U. and Nevanlinna O. (1987) Convergence of dynamic iteration methods for initial value problems. *SIAM J. Sci. Stat. Comput.* 8(4): 459–482.

[MN87b] Miekkala U. and Nevanlinna O. (1987) Sets of convergence and stability regions. *BIT* 27: 554–584.

[Pic93] Picard E. (1893) Sur l'application des méthodes d'approximations successives á l'etude de certaines équations différentielles ordinaires. *Journal de Mathématiques Pures et Appliquées* 9: 217–271.

17

Two Preconditioners for Saddle Point Problems with Penalty Term

Axel Klawonn

1 Introduction

Many problems in the engineering sciences lead to saddle point problems. Important examples are the Stokes equations of fluid dynamics, modeling the flow of an incompressible viscous fluid, and mixed formulations of problems from linear elasticity, e.g. for almost incompressible materials, plates, and beams; cf. Braess [Bra92] or Brezzi and Fortin [BF96]. These problems can be analyzed in the framework of saddle point problems with a penalty term.

In this article, we focus on the construction of preconditioned iterative method for certain saddle point problems with a penalty term. We present a block-diagonal and a block-triangular preconditioner in combination with appropriate Krylov space methods. This yields preconditioned iterative methods which have convergence rates that are bounded independently of the penalty and the discretization parameters. Details and proofs of the results can be found in [Kla97, Kla95a, Kla95b, Kla98]. Here, we only consider symmetric saddle point problems. For related work on the non-symmetric case, see Elman and Silvester [ES96], where the Oseen operator which is obtained by applying a Picard iteration to the Navier-Stokes equations is analyzed. Earlier work on block-diagonal preconditioners can be found in Rusten and Winther [RW92] and Silvester and Wathen [SW94]. A number of methods have been proposed for solving saddle point problems. For a list of references, see e.g. [Kla97, Kla98].

2 Saddle Point Problems with a Penalty Term

In this section, we give a brief overview over saddle point problems with a penalty term; cf. Braess [Bra92].

Let $(V, \|\cdot\|_V)$ and $(M, \|\cdot\|_M)$ be two Hilbert spaces, let M_c be a dense subspace of M, and let $a(\cdot,\cdot) : V \times V \to R$, $b(\cdot,\cdot) : V \times M \to R$, $c(\cdot,\cdot) : M_c \times M_c \to R$, be three bilinear forms. Additionally, we introduce V_0, a subspace of V, given by $V_0 := \{v \in V : b(v,q) = 0 \; \forall q \in M\}$. We assume that $a(\cdot,\cdot)$ is symmetric V_0-elliptic

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

and that $c(\cdot,\cdot)$ is symmetric M_c-positive semi-definite. Moreover, we assume $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ to be bounded. We consider the following problem:
Find $(u,p) \in V \times M_c$, such that

$$\begin{array}{rcrcll} a(u,v) & + & b(v,p) & = & <f,v> & \forall v \in V \\ b(u,q) & - & t^2 c(p,q) & = & <g,q> & \forall q \in M_c \quad t \in [0,1]. \end{array} \tag{1}$$

We denote by $X := V \times M_c$ the product space and by

$$\mathcal{A}(x,y) := a(u,v) + b(u,q) + b(v,p) - t^2 c(p,q),$$

$x = (u,p) \in X$, $y = (v,q) \in X$, the bilinear form of problem (1) on X. With the additional definition $\mathcal{F}(y) :=< f,v > + < g,q >$, we obtain an equivalent formulation of problem (1)

$$\mathcal{A}(x,y) = \mathcal{F}(y) \quad \forall y \in X. \tag{2}$$

We equip X with a new norm. We assume that we have an additional norm on M_c, i.e. $|||\cdot|||_M$, and introduce the new norm on X by

$$|||x||| := \|u\|_V + |||q|||_M \text{ for } x = (u,p) \in X.$$

If the bilinear form $c(\cdot,\cdot)$ is continuous on $M \times M$, we define $|||p|||_M := \|p\|_M$. Otherwise, $|||p|||_M$ is defined by $\|p\|_M + t|p|_c$, where $|p|_c := \sqrt{c(p,p)}$ is a semi-norm on M_c.

Theorem 1 *Let the following three assumptions be satisfied:*

1. *The continuous bilinear form $a(\cdot,\cdot)$ is symmetric and V_0-elliptic, i.e.*
$$\exists \alpha_0 > 0, \text{ such that } a(v,v) \geq \alpha_0 \|v\|_V^2 \quad \forall v \in V_0,$$

2. *The continuous bilinear form $b(\cdot,\cdot)$ fulfills an inf-sup condition, i.e.*
$$\exists \beta_0 > 0, \text{ such that } \inf_{q \in M} \sup_{v \in V} \frac{b(v,q)}{\|v\|_V \|q\|_M} \geq \beta_0,$$

3. *The bilinear form $c(\cdot,\cdot)$ is symmetric and M_c-positive semi-definite, i.e.*
$$c(q,q) \geq 0 \quad \forall q \in M_c.$$

Then, $\mathcal{A}(\cdot,\cdot)$ defines an isometric isomorphism $\mathcal{A} : X \to X'$ if in addition one of the following conditions is satisfied:

1) The bilinear form $c(\cdot,\cdot)$ is continuous on $M_c \times M_c$.
2) The bilinear form $a(\cdot,\cdot)$ is $V-$elliptic.

Under these assumptions $\mathcal{A}^{-1}$ is uniformly bounded for $t \in [0,1]$.

For a proof of this theorem, we refer the reader to Braess [Bra92], Section III.4.

All these results are also valid for suitable finite element spaces; see Braess [Bra92] or Brezzi and Fortin [BF96]. We then require, additionally, that the constants in Theorem 1 are independent of h. The continuity assumptions turn into uniform boundedness with respect to h; see, e.g., Braess [Bra92].

Discretizing (2) by mixed finite elements, we obtain a linear system of algebraic equations,

$$\mathcal{A}x = \mathcal{F},$$

where

$$\mathcal{A} := \begin{pmatrix} A & B^t \\ B & -t^2C \end{pmatrix} \in R^{n+m} \times R^{n+m}, \quad \mathcal{F} := \begin{pmatrix} f \\ g \end{pmatrix} \in R^{n+m}.$$

3 The Preconditioners

In this section, we present two different preconditioners for saddle point problems. The first is based on a block-diagonal structure, the second is a block-triangular preconditioner. Both approaches yield optimal convergence rates. To construct the preconditioners, we consider the discrete problem, using vectors and matrices, instead of functions and operators. Let us point out that it could have as well been presented in an abstract Hilbert space setting. With a slight abuse of notation, we also use, for simplicity, the same notation for the norms in both settings.

The Block-Diagonal Preconditioner

The block-diagonal preconditioner has the form

$$\hat{\mathcal{B}} := \begin{pmatrix} \hat{A} & O \\ O & \hat{C} \end{pmatrix} \in R^{n+m} \times R^{n+m}. \tag{3}$$

Here $\hat{A}$ and $\hat{C}$ satisfy certain ellipticity conditions, i.e. there exist positive constants a_0, a_1 and c_0, c_1, such that

$$\begin{aligned} a_0^2 \, \|u\|_V^2 &\leq u^t \hat{A} u \leq a_1^2 \, \|u\|_V^2 \quad \forall u \in R^n, \\ c_0^2 \, |||p|||_M^2 &\leq p^t \hat{C} p \leq c_1^2 \, |||p|||_M^2 \quad \forall p \in R^m. \end{aligned}$$

We assume the constants a_0, a_1, c_0, and c_1 to be independent of the critical parameters.

We use the block-diagonal preconditioner in combination with the conjugate residual method. To give a convergence estimate, it is our goal to give an upper bound for the spectral condition number of the preconditioned system $\kappa(\hat{\mathcal{B}}^{-1}\mathcal{A}) := \rho(\hat{\mathcal{B}}^{-1}\mathcal{A})\rho((\hat{\mathcal{B}}^{-1}\mathcal{A})^{-1})$, where $\rho(\cdot)$ is the spectral radius; cf. Hackbusch [Hac94], Theorem 9.5.13.

In the next theorem, we show that the spectral condition number can be uniformly bounded with respect to the penalty and discretization parameters; see Klawonn [Kla95a, Kla95b] for a detailed proof.

Theorem 2 *The condition number of $\hat{\mathcal{B}}^{-1}\mathcal{A}$ is bounded independently of the discretization and the penalty parameters, i.e.*

$$\kappa(\hat{\mathcal{B}}^{-1}\mathcal{A}) \leq \frac{C_1}{C_0}.$$

Here, C_0, C_1 are positive constants independent of the penalty and the discretization parameters.

Table 1 Iteration counts for exact solvers as preconditioners for A and C, $\nu = 0.3$.

Grid	20×10	40×20	60×30	80×40	100×50	120×60	140×70
CR	17	19	19	21	21	21	21
GMRES	9	10	10	10	10	10	10
BI-CGSTAB	5	5	6	6	6	6	6

The Block-Triangular Preconditioner

In this section, we consider a block-triangular preconditioner. We note that the results presented in this subsection are, in contrast to the previous subsection, restricted to saddle point problems where the blocks A and C are positive definite. This class of problems contains, e.g., Stokes equations or mixed formulations of linear elasticity but excludes certain beam and plate problems.

The preconditioned system is either of the form $\mathcal{A}\hat{\mathcal{B}}^{-1}$ or $\hat{\mathcal{B}}^{-1}\mathcal{A}$ where $\hat{\mathcal{B}}$ is the block–triangular preconditioner.

We use the following notation

$$\hat{\mathcal{B}}_U := \begin{pmatrix} \hat{A} & B^t \\ O & -\hat{C} \end{pmatrix} \in R^{n+m} \times R^{n+m}, \quad \hat{\mathcal{B}}_L := \begin{pmatrix} \hat{A} & O \\ B & -\hat{C} \end{pmatrix} \in R^{n+m} \times R^{n+m},$$

Here $\hat{A}$ and $\hat{C}$ are positive definite. We make the following assumptions on $\mathcal{A}$ and $\hat{\mathcal{B}}$: The matrix $\hat{A}$ is a good preconditioner for A, i.e.

$$\exists a_0, a_1 > 0 \quad a_0^2 u^t \hat{A} u \leq u^t A u \leq a_1^2 u^t \hat{A} u \quad \forall u \in R^n. \tag{4}$$

The constants a_0, a_1 should preferably be close to each other and be independent of the discretization parameters but there are also other interesting cases.

We also require that $\hat{C}$ is a good preconditioner for C, i.e.

$$\exists c_0, c_1 > 0 \quad c_0^2\, p^t \hat{C} p \leq p^t C p \leq c_1^2\, p^t \hat{C} p \quad \forall p \in R^m. \tag{5}$$

Under the additional assumption that

$$1 < a_0 \leq a_1, \tag{6}$$

which can always be achieved by an appropriate scaling, we can show that the spectrum of $\mathcal{A}\hat{\mathcal{B}}^{-1}$ stays bounded independently of the discretization and the penalty parameters.

Introduce the notation,

$$\mathcal{H} \;:=\; \begin{pmatrix} A - \hat{A} & O \\ O & \hat{C} \end{pmatrix}.$$

From (6), we see that $\mathcal{H}$ is positive definite. Since $A, \hat{A}, C$ are symmetric, it defines an inner product.

In the next theorem, we see that GMRES, using an inner product which is spectrally equivalent to the one defined by $\mathcal{H}^{-1}$, applied to the preconditioned system $\mathcal{A}\mathcal{B}_U^{-1}$ yields an optimal convergence rate. The proof uses that $\mathcal{A}\hat{\mathcal{B}}_U^{-1}$ is symmetric positive definite in the $\mathcal{H}^{-1}$–inner product. For details, see Klawonn [Kla95b, Kla98].

Table 2 Iteration counts for a two-level multigrid preconditioner with a standard V-cycle defining $\hat{A}$, and $\hat{C} = C$, and $\nu = 0.3$.

Grid	20×10	40×20	60×30	80×40	100×50	120×60	140×70
CR	20	23	24	26	26	26	26
GMRES	12	12	13	13	13	13	14
BI-CGSTAB	7	7	7	7	7	7	7

Table 3 Iteration counts for a two-level multigrid preconditioner with a standard V-cycle defining $\hat{A}$ and a one-level symmetric multiplicative overlapping Schwarz method with the minimal overlap of one node defining $\hat{C}$, and $\nu = 0.3$.

Grid	20×10	40×20	60×30	80×40	100×50	120×60	140×70
CR	20	23	24	26	26	26	26
GMRES	12	12	13	13	13	13	14
BI-CGSTAB	7	7	7	7	7	7	7

Theorem 3 *Let $\hat{\mathcal{H}}$ be a positive definite matrix, such that $\bar{C}_0^2 \quad \mathcal{H}^{-1} \leq \hat{\mathcal{H}}^{-1} \leq \bar{C}_1^2 \quad \mathcal{H}^{-1}$, where $\bar{C}_0, \bar{C}_1$ are positive constants independent of the discretization and penalty parameters. Then,*

$$\|r_n\|_{\hat{\mathcal{H}}^{-1}} \leq \frac{\bar{C}_1}{\bar{C}_0} \, 2 \, \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^n \quad \|r_0\|_{\hat{\mathcal{H}}^{-1}},$$

where r_n is the n-th residual of GMRES, $r_0 = b - \mathcal{A}\hat{\mathcal{B}}^{-1}x_0$ and $\kappa := \kappa(\mathcal{A}\hat{\mathcal{B}}^{-1}) \leq \frac{C_1}{C_0}$ is the condition number of $\mathcal{A}\hat{\mathcal{B}}^{-1}$ in the $\mathcal{H}^{-1}-$inner product.

We note that our convergence estimate only depends on the square root of the condition number of the preconditioned problem. Except for a leading factor, this estimate matches the standard estimate for the conjugate gradient method applied to positive definite symmetric problems.

There also exist convergence estimates for GMRES using the L_2-metric. Since these bounds are not uniform for the meshsize h, we refer to the more detailed discussion in Klawonn [Kla95b, Kla98]. We would like to point out that these theoretical non-uniform bounds are not sharp since the convergence rates obtained from the numerical experiments are uniformly bounded ; cf. Section 4.

4 Numerical Examples

In this section, we apply our block–preconditioners to the mixed formulation of planar, linear elasticity, cf., e.g., Brezzi and Fortin [BF96] or Klawonn [Kla97]. For simplicity,

Table 4 Iteration counts for exact solvers as preconditioners for A and C on a 80×40 grid.

ν	0.3	0.4	0.49	0.499	0.4999	0.49999	0.499999	0.5
CR	21	21	23	25	25	25	25	25
GMRES	10	11	12	12	12	12	12	12
BI-CGSTAB	6	7	7	7	7	7	7	7

Table 5 Iteration counts for a two-level multigrid method with a standard V-cycle defining $\hat{A}$ and $\hat{C} = C$ on a 80×40 grid.

ν	0.3	0.4	0.49	0.499	0.4999	0.49999	0.499999	0.5
CR	26	29	33	33	33	33	33	33
GMRES	13	14	15	15	15	15	15	15
BI-CGSTAB	7	7	7	7	7	7	7	7

we work with the following formulation

$$\begin{aligned} \mu\,(\nabla u, \nabla v)_0 + (\mathrm{div} v, p)_0 &= \; < f, v > \quad \forall v \in V := (H^1_\Gamma(\Omega))^2, \\ (\mathrm{div} u, q)_0 - \frac{1}{\lambda + \mu}\,(p, q)_0 &= \; 0 \qquad\qquad \forall q \in M := L_2(\Omega), \end{aligned}$$

with $H^1_\Gamma(\Omega) := \{v \in H^1(\Omega) : v_{|\Gamma_0} = 0\}$. Γ_0 is the part of the boundary where Dirichlet conditions are imposed and λ, μ are the Lamé parameters. All results shown are for mixed boundary conditions with $\Gamma_0 := \{x = (x_1, x_2) \in \partial\Omega : x_1 < -0.8\}$ and the region $[-1, 1] \times [-1, 1]$. We note that our model is mathematically equivalent to the full elasticity problem only in the case of Dirichlet conditions on the whole boundary.

For growing λ, the considered material becomes more incompressible. Instead of using the Lamé constants λ and μ, we can also work with Young's elasticity modulus E and the Poisson ratio ν. These parameters are related to each other as follows

$$\lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}, \quad \mu = \frac{E}{2(1+\nu)}. \tag{7}$$

The relation between the penalty parameter t and the Poisson ratio ν is given by $t := (1+\nu)(1-2\nu)/(E\nu)$. Without loss of generality, we set $E = 1$. We discretize by a $Q_1(h) - Q_1(2h)$ macro-element, i.e. we use piecewise bilinear polynomials on quadrilaterals on a grid with mesh size h for the displacements u and piecewise bilinear polynomials on quadrilaterals with mesh size $2h$ for the Lagrange multiplier p. For a proof that the inf-sup condition of B holds for this element; see Girault and Raviart [GR86] or Brezzi and Fortin [BF96].

All computations were carried out on a SUN SPARC 10 workstation using the numerical software package PETSc 1.0; cf. Balay, Gropp, Curfman McInnes, and Smith [BGMS96]. The initial guess is 0, and the stopping criterion is $\|r_k\|_2 / \|r_0\|_2 < 10^{-5}$, where r_k is the k-th residual.

Table 6 Iteration counts for an exact solver as $\hat{A}$ and a one-level symmetric multiplicative overlapping Schwarz method with the minimal overlap of one node as $\hat{C}$ on a 80 × 40 grid.

ν	0.3	0.4	0.49	0.499	0.4999	0.49999	0.499999	0.5
CR	21	21	23	25	25	25	25	25
GMRES	10	11	12	12	12	12	12	12
BI-CGSTAB	6	7	7	7	7	7	7	7

Table 7 Iteration counts for a two-level multigrid method with a standard V-cycle as $\hat{A}$ and a one-level symmetric multiplicative overlapping Schwarz method with the minimal overlap of one node as $\hat{C}$ on a 80 × 40 grid.

ν	0.3	0.4	0.49	0.499	0.4999	0.49999	0.499999	0.5
CR	26	29	33	33	33	33	33	33
GMRES	13	14	15	15	15	15	15	15
BI-CGSTAB	7	7	7	7	7	7	7	7

We give numerical results for different Krylov space methods. We use the conjugate residual method (CR) in combination with the block–diagonal preconditioner and GMRES and BI-CGSTAB with the block-triangular one. We report on results with a version of GMRES without restarts but we also ran a version with restart every 10 iterations. The number of iterations for this latter version was always just one or two larger than without restart. We use right-oriented preconditioning with $\hat{\mathcal{B}}_U^{-1}$ for GMRES and we only use the L_2- rather than the $\hat{\mathcal{H}}^{-1}-$metric. Experiments were also carried out with a left-oriented preconditioner $\hat{\mathcal{B}}_L^{-1}$ and BI-CGSTAB. This latter method has the advantage of being based on a short term recurrence but it is not covered by our theory. As is shown in our experiments, there is no appreciable difference in the number of matrix-vector products of the different methods and the numerical results suggest that the number of iterations is bounded independently of the critical parameters h and t. Although we would like to point out that GMRES requires more inner products and more storage than BI-CGSTAB.

We note that in all of our experiments the block-diagonal preconditioner almost always needs about twice as many matrix-vector products than the block-triangular preconditioner. Whereas the latter is only slightly more expensive when used with a short term recurrence method.

To see how the Krylov space methods behave under the best of circumstances, we first conducted some experiments using exact solvers, i.e. $\hat{A} = A$ and $\hat{C} = C$; see Tables 1 and 4.

In another series of experiments, we use different preconditioners for A and C. We present results with a two-level multigrid preconditioner with a V-cycle including one pre- and one post-smoothing symmetric Gauss-Seidel step defining $\hat{A}$, and a one-level symmetric multiplicative overlapping Schwarz method with the minimal overlap of one node as $\hat{C}$; see Tables 2, 3, 5, 6, 7.

REFERENCES

[BF91] Brezzi F. and Fortin M. (1991) *Mixed and Hybrid Finite Element Methods.* Springer-Verlag.

[BGMS96] Balay S., Gropp W., McInnes L. C., and Smith B. (April 1996) PETSc World Wide Web home page. http://www.mcs.anl.gov/petsc/petsc.html.

[Bra92] Braess D. (1992) *Finite Elemente.* Springer-Verlag.

[ES96] Elman H. and Silvester D. (January 1996) Fast Nonsymmetric Iterations and Preconditioning for Navier-Sokes Equations. *SIAM J. Sci. Comp.* 17(1): 33–46.

[GR86] Girault V. and Raviart P.-A. (1986) *Finite Element Methods for Navier-Stokes Equations.* Springer-Verlag.

[Hac94] Hackbusch W. (1994) *Iterative Solution of Large Sparse Systems of Equations.* Springer, New York.

[Kla95a] Klawonn A. (April 1995) An optimal preconditioner for a class of saddle point problems with a penalty term, Part II: General theory. Technical Report 14/95, Westfälische Wilhelms-Universität Münster, Germany. Also available as Technical Report 683 at the Courant Institute of Mathematical Sciences, New York University.

[Kla95b] Klawonn A. (1995) *Preconditioners for Indefinite Problems.* PhD thesis, Westfälische Wilhelms-Universität Münster.

[Kla97] Klawonn A. (November 1997) An optimal preconditioner for a class of saddle point problems with a penalty term. *SIAM J. Sci. Comp.* 18(6). To appear.

[Kla98] Klawonn A. (January 1998) Block-Triangular Preconditioners for Saddle Point Problems with a Penalty Term. *SIAM J. Sci. Comp.* 19(1). To appear.

[RW92] Rusten T. and Winther R. (1992) A preconditioned iterative method for saddle point problems. *SIAM J. Matrix Anal. Appl.* 13: 887–904.

[SW94] Silvester D. and Wathen A. (1994) Fast iterative solutions of stabilised Stokes systems Part II: Using general block preconditioners. *SIAM J. Numer. Anal.* 31(5): 1352–1367.

18

A Domain Decomposition Method for Micropolar Fluids

Piotr Krzyżanowski

1 Introduction

In this paper, we consider a mixed finite element discretization of the following system of partial differential equations with Dirichlet boundary conditions:

$$\begin{cases} -(\nu+\nu_1)\Delta u + (u\cdot\nabla)u + \nabla p = 2\nu_1 \operatorname{curl}\omega + f & \text{in } \Omega, \\ \operatorname{div} u = 0 & \text{in } \Omega, \\ -c_1\Delta\omega + (u\cdot\nabla)\omega - c_2\nabla\operatorname{div}\omega + 4\nu_1\omega = 2\nu_1\operatorname{curl} u + g & \text{in } \Omega, \\ u = u_0 & \text{on } \partial\Omega, \\ \omega = \omega_0 & \text{on } \partial\Omega \end{cases} \tag{1}$$

This system is a mathematical model of stationary flow of a viscous micropolar fluid, which describes the motion of solid particle suspension in a liquid. Such model is also a basis for more complicated ones used in applied sciences, for example in the theory of lubrication [BL95], [Kho90] or in the theory of blood flow [P$^+$74].

The unknowns are the velocity vector u, the pressure p and the internal microrotation vector ω. We denote the external force and the angular momentum force by f and g, respectively. The usual (constant) kinematic viscosity is denoted by $\nu > 0$, while other positive constants ν_1, c_1 and c_2 are additional viscosities introduced by the field of internal rotation ω.

Existence and uniqueness theorems for (1) are proved in [Luk88]. Here, we provide their discrete counterparts for the mixed finite element discretization of (1). The nonlinear discrete problem is then solved using the Newton's method. Each iteration step requires solution of a linear system with a nonsymmetric indefinite matrix, which is ill conditioned with respect to the mesh size h.

We propose and analyse a preconditioning method for the linear system, based on a block diagonal preconditioner. Our goal is to make it possible to reuse the methods already existing for simpler problems, like for the Poisson equation. Since the theory and methods for preconditioning the discrete Laplacian are well developed, our preconditioner can be easily constructed and implemented, using, for example, an efficient domain decomposition preconditioner.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

The preconditioned system *is* symmetric and positive definite with respect to some auxiliary scalar product, so standard iterative methods, like conjugate gradient method, can be used for this system. Each step of CG method requires solution of three smaller, independent problems of small computational cost.

Block diagonal preconditioners for Stokes-like problems have been considered by many authors before, see, for example, [D'y87], [BP88], [BP90], [RW92], [ES94], [SW94] or [Kla96]. However, our analysis of the preconditioned system relies neither on the symmetry nor the positive definiteness of the matrix.

Notation

Throughout the paper we assume that Ω is an open, bounded polyhedron in R^3 with Lipschitz continuous boundary. The differential operators $\Delta, \nabla, \mathrm{curl}, \mathrm{div}$, appearing in (1), are defined in a standard way, see [GR86].

We use several function spaces whose properties are described, for example, in [Ada75]. By $H^k(\Omega)$ we denote the usual Sobolev spaces, identifying $H^0(\Omega)$ with the $L^2(\Omega)$ space of square integrable functions. The standard norm in $H^k(\Omega)$ is denoted by $||\cdot||_k$, while the seminorm by $|\cdot|_k$. $H^1_0(\Omega)$ denotes the subspace of $H^1(\Omega)$ of functions whose traces on $\partial\Omega$ are equal to zero, while $L^2_0(\Omega)$ is the subspace of $L^2(\Omega)$, defined as $L^2_0(\Omega) = \{w \in L^2(\Omega) : \int_\Omega w = 0\}$.

For a positive integer N, we denote the inner product in $[L^2(\Omega)]^N = L^2(\Omega) \times L^2(\Omega) \ldots \times L^2(\Omega)$ by

$$(u, v) := \sum_{i=1}^{N} \int_\Omega u_i \, v_i \, dx.$$

For the inner product in $[H^1_0(\Omega)]^N$ we use

$$((u, v)) := \sum_{j=1}^{N} (\nabla u_j, \nabla v_j) = \sum_{j=1}^{N} \sum_{i=1}^{N} \int_\Omega \frac{\partial u_j}{\partial x_i} \frac{\partial v_j}{\partial x_i}.$$

By "C" we denote a generic positive constant which, if necessary, we shall distinguish by subscripts. Where there is no risk of confusion, we shall write H^k, H^1_0, L^2_0 instead of $H^k(\Omega)$, $H^1_0(\Omega)$, $L^2_0(\Omega)$ and use the same symbols for N-fold products of such spaces.

The Discrete Problem

We pose our original problem (1) in a variational form, using the following function spaces:

$$V := [H^1_0(\Omega)]^3, \qquad W := L^2_0(\Omega).$$

In the rest of the paper we shall assume that the data for problem (1) satisfy $f, g \in [L^2(\Omega)]^3$ and the boundary conditions on u, ω are homogeneous.

We cover $\bar{\Omega}$ with a quasi-uniform triangulation [Cia91] $\mathcal{T}_h$, dividing $\bar{\Omega}$ into tetrahedra K:

$$\bigcup_{K \in \mathcal{T}_h} K = \bar{\Omega},$$

with the mesh parameter h. We make standard assumption that at least one vertex of each $K \in \mathcal{T}_h$ lies inside Ω.

For approximation of the velocity u and pressure p we shall use the Taylor – Hood finite spaces V_h, W_h (see, e.g. [BF91]),

$$V_h = \{v \in V \cap C(\bar{\Omega}) : v\big|_K \in P_2(K) \quad \forall K \in \mathcal{T}_h\},$$

and

$$W_h = \{w \in W \cap C(\bar{\Omega}) : w\big|_K \in P_1(K) \quad \forall K \in \mathcal{T}_h\}.$$

The microrotation field ω is approximated in the same space V_h as the velocity. It is well known that V_h and W_h satisfy the *inf-sup* condition.

The mixed variational formulation of the approximate problem (1) in the finite element spaces $V_h \subset V$, $W_h \subset W$ is as follows:

Problem 1.1 *Find* $(u_h, p_h, \omega_h) \in V_h \times W_h \times V_h$, *such that*

$$\begin{cases} (\nu + \nu_1)(\nabla u_h, \nabla v) + d(u_h,\, u_h,\, v) - (p_h, \operatorname{div} v) = 2\nu_1(\operatorname{curl} \omega_h, v) + (f, v), \\ (\operatorname{div} u_h, q) = 0, \\ c_1(\nabla \omega_h, \nabla \xi) + d(u_h,\, \omega_h,\, \xi) + c_2(\operatorname{div} \omega_h, \operatorname{div} \xi) + 4\nu_1(\omega_h, \xi) \\ \qquad\qquad = 2\nu_1(\operatorname{curl} u_h, \xi) + (g, \xi), \end{cases} \tag{2}$$

for all (v, q, ξ) *in* $V_h \times W_h \times V_h$.

Here, $d(\cdot, \cdot, \cdot)$ denotes the convective term, defined either as

$$d_1(u,\, v,\, w) := ((u \cdot \nabla)v, w) = \sum_{i,j=1}^{3} \int_\Omega u_i \frac{\partial v_j}{\partial x_i} w_j,$$

or, following [Tem79],

$$d_2(u,\, v,\, w) := \frac{1}{2}\Big(((u \cdot \nabla)v, w) - ((u \cdot \nabla)w, v)\Big)$$

for any $u, v, w \in [H^1(\Omega)]^3$. Note that if $\operatorname{div} u = 0$ then $d_1(u,\, \cdot,\, \cdot) \equiv d_2(u,\, \cdot,\, \cdot)$. The form $d_2(\cdot,\, \cdot,\, \cdot)$ is by the definition skew-symmetric with respect to the last two arguments (which reflects the skew-symmetry of $((u \cdot \nabla)v, w)$ on the solution u of (1)).

2 Existence and Uniqueness Results

We begin with a general existence result for the case when the form $d(\cdot,\, \cdot,\, \cdot)$ is equal to $d_2(\cdot,\, \cdot,\, \cdot)$.

Theorem 2.1 *For any* $f, g \in L^2$ *and any positive* ν, ν_1, c_1, c_2 *there exists at least one triple* $(u_h, p_h, \omega_h) \in V_h \times W_h \times V_h$ *which solves the discrete nonlinear system (2). Moreover, the solution is unique, provided that the data* f, g *are sufficiently small with respect to* ν, ν_1, c_1, c_2.

Remark 2.1 *The "small data" assumption in Theorem 2.1 reflects similar requirements of the uniqueness statement for the continuous case, see [Luk88].*

The next theorem is valid for $d(\cdot,\cdot,\cdot) \equiv d_1(\cdot,\cdot,\cdot)$ or $d_2(\cdot,\cdot,\cdot)$ and provides a generalization of the discrete Navier–Stokes local uniqueness and approximation result of [GR86] for the discrete micropolar equations.

Theorem 2.2 *Let us set $\lambda = (\nu + \nu_1)^{-1}$. Let Λ be a compact interval in R_+ and assume that $\{(\lambda, (u(\lambda), \lambda p(\lambda), \omega(\lambda)) : \lambda \in \Lambda\}$ is a branch of nonsingular solutions of (1) such that $u(\lambda) \in H^{l+1}$, $p(\lambda) \in H^l$, $\omega(\lambda) \in H^{l+1}$ for $l = 1$ or $l = 2$ and for all $\lambda \in \Lambda$. Then there exists h_0 (small enough), such that for $h \leq h_0$ there exists a unique smooth function $\lambda \in \Lambda \to (u_h(\lambda), \lambda p_h(\lambda), \omega_h(\lambda)) \in V_h \times W_h \times V_h$ such that:*

(i) $\{(\lambda, (u_h(\lambda), \lambda p_h(\lambda), \omega_h(\lambda)) : \lambda \in \Lambda\}$ is a branch of nonsingular solutions of Problem 1.1,

(ii) there exists $C > 0$, independent of h and λ, such that for all $\lambda \in \Lambda$

$$|u_h(\lambda) - u(\lambda)|_1 + |\lambda| ||p_h(\lambda) - p(\lambda)||_0 + |\omega_h(\lambda) - \omega(\lambda)|_1 \\ \leq Ch^l(||u(\lambda)||_{l+1} + |\lambda| ||p(\lambda)||_l + ||\omega(\lambda)||_{l+1})$$

For the proofs of Theorem 2.1 and Theorem 2.2 we refer the reader to [Krz96]. Existence and local uniqueness results stated in Theorem 2.1 and in Theorem 2.2 can be easily extended to other conforming finite elements satisfying the inf-sup condition.

3 A Preconditioning Method for Newton's Iteration Step

In this section we propose and analyse a preconditioning method for one step of Newton's method for Problem 1.1.

Newton's algorithm. Given $(u_h^n, p_h^n, \omega_h^n) \in V_h \times W_h \times V_h$, find $(u_h^{n+1}, p_h^{n+1}, \omega_h^{n+1}) \in V_h \times W_h \times V_h$, which satisfies

$$\begin{cases} (\nu + \nu_1)(\nabla u_h^{n+1}, \nabla v) + d(u_h^n,\, u_h^{n+1},\, v) + d(u_h^{n+1},\, u_h^n,\, v) - (p_h, \operatorname{div} v) \\ \qquad = 2\nu_1(\operatorname{curl} \omega_h^{n+1}, v) + d(u_h^n,\, u_h^n,\, v) + (f, v), \\ (\operatorname{div} u_h^{n+1}, q) = 0, \\ c_1(\nabla \omega_h^{n+1}, \nabla \xi) + d(u_h^n,\, \omega_h^{n+1},\, \xi) + d(u_h^{n+1},\, \omega_h^n,\, \xi) + c_2(\operatorname{div} \omega_h^{n+1}, \operatorname{div} \xi) \\ \qquad + 4\nu_1(\omega_h^{n+1}, \xi) = 2\nu_1(\operatorname{curl} u_h^{n+1}, \xi) + d(u_h^n,\, \omega_h^n,\, \xi) + (g, \xi) \end{cases} \tag{3}$$

for all $(v, q, \xi) \in V_h \times W_h \times V_h$.

Actually, we are dealing with a family of such problems, indexed by the mesh parameter h. Under assumptions as in Theorem 2.2 the Newton's method is locally quadratically convergent to the solution of the discrete system. The rate of convergence is affected by the parameter $\lambda = (\nu + \nu_r)^{-1}$, but is independent (see [Krz96]) of the mesh parameter h.

We are going to analyse a preconditioning method for these problems so that the resulting problem is given by a symmetric positive definite operator whose condition number is independent of h. Let us denote for short $(u_h^{n+1}, p_h^{n+1}, \omega_h^{n+1})$ by (u, p, ω). Define linear operators A, B, C, T_A, T_B by variational identities for all $u, v \in V_h$ and

$w \in W_h$:

$$\begin{array}{ll} A : V_h \to V_h', & \langle\langle Au, v\rangle\rangle = (\nu + \nu_1)(\nabla u, \nabla v) + d(u_h^n,\, u,\, v) + d(u,\, u_h^n,\, v), \\ B : V_h \to W_h, & \langle\langle Bu, w\rangle\rangle = -(\operatorname{div} u, w), \quad B' : W_h \to V_h', \langle\langle B'w, u\rangle\rangle = -(\operatorname{div} u, w), \\ C : V_h \to V_h', & \langle\langle Cu, v\rangle\rangle = c_1(\nabla\omega, \nabla\xi) + d(u_h^n,\, \omega,\, \xi) + c_2(\operatorname{div}\omega, \operatorname{div}\xi) + 4\nu_1(\omega, \xi), \\ T_A : V_h \to V_h', & \langle\langle T_A u, v\rangle\rangle = -2\nu_1(\operatorname{curl}\omega, v), \\ T_C : V_h \to V_h', & \langle\langle T_C u, v\rangle\rangle = -2\nu_1(\operatorname{curl} u, \xi) + d(u,\, \omega_h^n,\, \xi). \end{array}$$

The dual pairing between V_h and V_h', or W_h and W_h', respectively, is denoted by $\langle\langle\cdot,\cdot\rangle\rangle$. We use the same symbol for these two different pairings, since its meaning will be always clear form the context. Then we can express (3) in an operator form:

Problem 3.1 *For $\mathcal{F} = (\phi, \psi, \varphi) \in V_h' \times W_h' \times V_h'$, find $(u, p, \omega) \in V_h \times W_h \times V_h$ such that*

$$\mathcal{M}\begin{pmatrix} u \\ p \\ \omega \end{pmatrix} \equiv \begin{pmatrix} A & B' & T_A \\ B & 0 & 0 \\ T_C & 0 & C \end{pmatrix}\begin{pmatrix} u \\ p \\ \omega \end{pmatrix} = \mathcal{F}.$$

The following lemma is a consequence of Theorem 2.2, see [Krz96].

Lemma 3.1 *Suppose that the assumptions of Theorem 2.2 hold with sufficiently small h_0. In addition, let us assume that $(u_h^n, p_h^n, \omega_h^n)$ is close enough to the solution of Problem 1.1. Then for any $\mathcal{F} \in V_h' \times W_h' \times V_h'$ there exists a unique solution (u, p, ω) of Problem 3.1 and*

$$||u||_1 + ||p||_0 + ||\omega||_1 \le C(||\phi||_{V_h'} + ||\psi||_{W_h'} + ||\varphi||_{V_h'})$$

with C independent of h.

With the inner product $((\cdot,\cdot))$ in V_h we associate the discrete Laplace operator $-\Delta_h : V_h \to V_h'$, defined by $\langle\langle -\Delta_h u, v\rangle\rangle \equiv ((u, v))$. We also define canonical mapping $J : W_h \to W_h'$ (the "mass matrix" operator), $\langle\langle Jp, q\rangle\rangle \equiv (p, q)$.

Let $A_0 : V_h \to V_h'$ be a good preconditioner for the discrete Laplace operator $-\Delta_h$, so that

(i) $\langle\langle A_0 u, v\rangle\rangle = \langle\langle A_0 v, u\rangle\rangle$ for all $u, v \in V_h$,

(ii) there exist constants $\alpha_1, \alpha_2 > 0$, independent of h, such that

$$\alpha_1\langle\langle -\Delta_h u, u\rangle\rangle \le \langle\langle A_0 u, u\rangle\rangle \le \alpha_2\langle\langle -\Delta_h u, u\rangle\rangle \qquad (4)$$

for all $u \in V_h$,

(iii) A_0^{-1} is easy to apply.

Likewise, we introduce (cf. [Kla96]) a good preconditioner for the "mass matrix" operator, $J_0 : W_h \to W_h'$, i.e.

(iv) $\langle\langle J_0 p, q\rangle\rangle = \langle\langle J_0 q, p\rangle\rangle$ for all $p, q \in W_h$,

(v) there exist constants $\beta_1, \beta_2 > 0$, independent of h, such that

$$\beta_1\langle\langle Jp, p\rangle\rangle \le \langle\langle J_0 p, p\rangle\rangle \le \beta_2\langle\langle Jp, p\rangle\rangle \qquad (5)$$

for all $p \in W_h$,

(vi) J_0^{-1} is easy to apply.

In the implementation, efficient preconditioners A_0 and J_0 may be obtained using domain decomposition methods.

Introducing a block diagonal operator matrix

$$\mathcal{M}_0 = \begin{pmatrix} A_0 & 0 & 0 \\ 0 & J_0 & 0 \\ 0 & 0 & A_0 \end{pmatrix}, \tag{6}$$

we define a preconditioned version of problem (3).

Problem 3.2 *Find $(u, p, \omega) \in V_h \times W_h \times V_h$ such that*

$$\mathcal{M}_0^{-1}\mathcal{M}'\mathcal{M}_0^{-1}\mathcal{M} \begin{pmatrix} u \\ p \\ \omega \end{pmatrix} = \mathcal{M}_0^{-1}\mathcal{M}'\mathcal{M}_0^{-1}\mathcal{F}.$$

Lemma 3.2 *The operator $\mathcal{P} = \mathcal{M}_0^{-1}\mathcal{M}'\mathcal{M}_0^{-1}\mathcal{M}$ is self-adjoint with respect to the auxiliary scalar product $[\cdot,\cdot]$ defined as*

$$\left[\begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} v \\ q \\ z \end{pmatrix} \right] \equiv \langle\langle A_0 u, v \rangle\rangle + \langle\langle J_0 p, q \rangle\rangle + \langle\langle A_0 \omega, z \rangle\rangle. \tag{7}$$

The main result of this section is an estimate of the condition number of the operator $\mathcal{P}$ in the norm induced by $[\cdot,\cdot]$.

Theorem 3.3 *Let $\mathcal{P}$ be defined as in Lemma 3.2. Suppose Lemma 3.1 holds and A_0, J_0 satisfy assumptions (i) – (vi) of this section. Then there exist positive constants m_1, m_2, independent of h, such that*

$$m_1 \left[\begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix} \right] \le \left[\mathcal{P} \begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix} \right] \le m_2 \left[\begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix} \right]$$

for any $(u, p, \omega) \in V_h \times W_h \times V_h$.

Proof. We have

$$\begin{aligned} \left[\mathcal{P} \begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix} \right] &= \langle\langle Au + B'p + T_A\omega, A_0^{-1}(Au + B'p + T_A\omega) \rangle\rangle + \langle\langle Bu, J_0^{-1}Bu \rangle\rangle \\ &\quad + \langle\langle T_C u + C\omega, A_0^{-1}(T_C u + C\omega) \rangle\rangle. \end{aligned} \tag{8}$$

Since there exist constants $\gamma_1, \gamma_2 > 0$, and $\delta_1, \delta_2 > 0$, independent of h, such that

$$\begin{aligned} \gamma_1 ||\phi||_{V_h'}^2 \le \langle\langle \phi, A_0^{-1}\phi \rangle\rangle \le \gamma_2 ||\phi||_{V_h'}^2 \qquad \forall \phi \in V_h', \\ \delta_1 ||q||_0^2 \le \langle\langle q, J_0^{-1} q \rangle\rangle \le \delta_2 ||q||_0^2 \qquad \forall q \in W_h', \end{aligned} \tag{9}$$

we obtain

$$\left[\mathcal{P}\begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix}\right] \geq \gamma_1 ||Au + B'p + T_A\omega||^2_{V_h'} + \delta_1 ||Bu||^2_0 + \gamma_1 ||T_C u + C\omega||^2_{V_h'},$$

which by Lemma 3.1 together with (4) and (5) yields the lower bound

$$\left[\mathcal{P}\begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix}\right] \geq C \left[\begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix}\right].$$

Similarly we can establish the upper bound. Indeed, from (9) together with (8) we have

$$\begin{aligned}\left[\mathcal{P}\begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix}\right] &\leq \gamma_2 ||Au + B'p + T_A\omega||^2_{V_h'} + \delta_2 ||Bu||^2_0 + \gamma_2 ||T_C u + C\omega||^2_{V_h'} \\ &\leq C(||Au||^2_{V_h'} + ||B'p||^2_{V_h'} + ||T_A\omega||^2_{V_h'} + ||Bu||^2_0 + ||T_C u||^2_{V_h'} + ||C\omega||^2_{V_h'}).\end{aligned}$$

Obviously, each of the operators A, B, C, T_A, T_B is bounded from above independently of h. Estimating each term in the sum, we get

$$\left[\mathcal{P}\begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix}\right] \leq C \left[\begin{pmatrix} u \\ p \\ \omega \end{pmatrix}, \begin{pmatrix} u \\ p \\ \omega \end{pmatrix}\right],$$

which completes the proof.

4 Remarks

The resulting system can be solved by the conjugate gradient method since its matrix is symmetric and positive definite. By Theorem 3.3, the number of iterations required for reducing the residual by a given factor is independent of h. As it has been pointed out in [BP88], computing the inner product during the CG step requires only one solution of a system with the operator $\mathcal{M}_0$.

Many authors contributed to the area of numerical solution of saddle point problems, see for example [D'y87], [BP88], [BP90], [RW92], [ES94], [SW94], [Kla96], addressing mostly (if not exclusively) the symmetric operator case. The idea of symmetrizing the saddle point system with the aid of a preconditioner for the Laplacian has been considered previously in, for example, [D'y87] and [BP90]. However, our analysis remains also valid for nonsymmetric operators.

In the case of $v_r = c_1 = c_2 = 0$ our system reduces to Newton's linearization of the Navier-Stokes equations, therefore our preconditioner applies also to this particular case. Moreover, this preconditioning method generalizes to the case of abstract saddle point equations with nonsymmetric, indefinite diagonal part. Different preconditioning methods for these problems will be analysed in a forthcoming paper.

REFERENCES

[Ada75] Adams R. (1975) *Sobolev spaces.* Academic Press, New York.

[BF91] Brezzi F. and Fortin M. (1991) *Mixed and Hybrid Finite Element Methods.* Springer-Verlag, New York.

[BL95] Bayada G. and Łukaszewicz G. (1995) On micropolar fluids in the theory of lubrication. Rigorous derivation of an analogue of the Reynolds equation. Technical Report RW 95-12, Institute of Applied Mathematics and Mechanics, Warsaw University.

[BP88] Bramble J. and Pasciak J. (1988) A preconditioning technique for indefinite problems resulting from mixed approximation of elliptic problems. *Math. Comp.* 50: 1–17.

[BP90] Bramble J. and Pasciak J. (1990) A domain decomposition technique for Stokes problems. *App. Num. Math.* 6: 251–261.

[Cia91] Ciarlet P. (1991) *Basic Error Estimates for Elliptic Problems, Handbook of Numerical Analysis*, volume II, Finite Element Methods (Part I). Elsevier Science Publishers B.V. (North-Holland).

[D'y87] D'yakonov E. (1987) On iterative methods with saddle operators. *Soviet Math. Dokl.* 35(1): 166–170.

[ES94] Elman H. and Silvester D. (1994) Fast nonsymmetric iterations and preconditioning for Navier–Stokes equations. *To appear* .

[GR86] Girault V. and Raviart P. (1986) *Finite Element Method for Navier–Stokes Equations. Theory and Algorithms.* Springer-Verlag, Berlin, Heidelberg, New York.

[Kho90] Khonsari M. (1990) On the self-excited orbits of a journal bearing in a sleeve bearing lubricated with micropolar fluids. *Acta Mechanica* 87: 235–244.

[Kla96] Klawonn A. (1996) *Preconditioners for Indefinite Problems.* PhD thesis, Universität Münster, Germany.

[Krz96] Krzyżanowski P. (1996) Mixed finite element method for micropolar fluids. Technical Report RW 96-09, Institute of Applied Mathematics and Mechanics, Warsaw University.

[Luk88] Łukaszewicz G. (1988) On stationary flows of asymmetric fluids. *Rend. Acc. Naz. Sci. XL* XII(106): 35–44.

[P$^+$74] Popel A. *et al.* (1974) A continuum model of blood flow. *Biorheology* XI: 427–437.

[RW92] Rusten T. and Winther R. (1992) A preconditioned iterative method for saddle point problems. *SIAM J. Matr. Anal. Appl.* 13: 887–904.

[SW94] Silvester D. and Wathen A. (1994) Fast iterative solution of stabilized Stokes systems, Part II: Using general block preconditioners. *SIAM J. Numer. Anal.* 31(5): 1352–1367.

[Tem79] Temam R. (1979) *Navier – Stokes equations. Theory and numerical analysis.* North-Holland, Amsterdam New York Oxford.

19

A Note on Domain Decomposition of Singularly Perturbed Elliptic Problems

A. Auge, A. Kapurkin, G. Lube and F. C. Otto

1 Introduction

Considerable progress can be observed in the design and analysis of parallelizable domain decomposition methods for singularly perturbed elliptic problems (see references). The purpose of this paper is to report on some results and problems with two domain decomposition methods for the advection–diffusion–reaction model.

Reasonable results are now available for *overlapping* Schwarz methods. In particular, the overlap can be minized in the singularly perturbed case using certain exponential decay of Dirichlet data in overlap regions. In Sec. 3 we consider a modified Schwarz method which is easy to parallelize in case of a simple geometry. We derive error estimates in the continuous case which can be extended to the discrete case.

Non–overlapping methods are better suited for parallel implementation. The problem consists in deriving appropriate interface conditions. We consider in Sec. 4 a method with an adaptive interface condition and discuss recent variants. Strong convergence is proven for the continuous method. We obtained a robust behaviour of both methods for two– and three–dimensional test problems using stabilized Galerkin finite element methods as the basic discretization.

2 Preliminaries

Consider the following Dirichlet problem in a bounded domain $\Omega \subset \mathbf{R}^d,\ d = 2, 3$ with Lipschitzian boundary $\partial\Omega$:

$$L_\epsilon u := -\epsilon \Delta u + (\boldsymbol{a} \cdot \nabla) u + cu = f \quad \text{in } \Omega; \qquad u = 0 \quad \text{on } \partial\Omega \tag{1}$$

Of particular interest are singularly perturbed problems with $\|\boldsymbol{a}\|_\infty \gg \epsilon$ (advection dominated case) or $\|c\|_\infty \gg \max\{\|\boldsymbol{a}\|_\infty; \epsilon\}$ (reaction dominated case). The latter case appears e.g. in an implicit time discretization method. We assume sufficiently smooth

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

data of the problem satisfying

(H.1) $c(x) \geq c_0 > 0, \; x \in \Omega; \qquad \nabla \cdot \boldsymbol{a} = 0.$

Remark 2.1. The condition $c(x) \geq c_0 > 0$ can be always guaranteed if all characteristic curves, the solutions of $dx(\tau)/d\tau = \boldsymbol{a}(x(\tau)), x(0) \in \overline{\Omega}$, leave $\overline{\Omega}$ in finite time. The incompressibility condition on $\boldsymbol{a}$ is not essential. □

The basic variational problem is: Find $u \in W \equiv H_0^1(\Omega)$, such that for all $v \in W$

$$B^G(u,v) := (\epsilon \nabla u, \nabla v)_\Omega + (\boldsymbol{a} \cdot \nabla u, v)_\Omega + (cu, v)_\Omega = L^G(v) := (f, v)_\Omega \tag{2}$$

Let $\mathcal{T} = \{K\}$ be an admissible triangulation and $W^h \subset W$ a finite element space of piecewise polynomials of degree $k \geq 1$. The (unusual) stabilized Galerkin method [FFLR96] for problem (1) is to find $U \in W^h$ such that

$$\begin{aligned} B^{SG}(U,v) &= L^{SG}(v) \quad \forall v \in W^h, \\ B^{SG}(u,v) &:= B^G(u,v) - \textstyle\sum_K \sigma_K (L_\epsilon u, L_\epsilon^* v)_K, \\ L^{SG}(v) &:= L^G(v) - \textstyle\sum_K \sigma_K (f, L_\epsilon^* v)_K, \end{aligned} \tag{3}$$

where L_{ϵ^*} denotes the adjoint operator to L_ϵ and σ_K are suitably chosen parameters. For a subdomain $D \subset \Omega$ set $H_{0,\Omega}^1(D) := W \cap H^1(D)$. $B_D^{SG}(\cdot,\cdot)$ and $L_D^{SG}(\cdot,\cdot)$ are the obvious restrictions of $B^{SG}(\cdot,\cdot)$ and $L^{SG}(\cdot,\cdot)$ to D if D consists of finite elements in $\mathcal{T}$. Then define

$$V^h(D) := H^1(D) \cap W^h; \quad V_0^h(D) := H_0^1(D) \cap V^h(D).$$

By ∂D^+, ∂D^- and ∂D^0 we denote the outflow, inflow and characteristic parts of ∂D where the scalar product $\boldsymbol{a} \cdot \boldsymbol{\nu}_D$ with the outer normal $\boldsymbol{\nu}_D$ is positive, negative or zero, respectively.

Furthermore we introduce a nonoverlapping admissible partition $\overline{\Omega} = \cup_i \overline{\Omega}_i$ with Lipschitz $\partial\Omega_i$ which aligns with the triangulation $\mathcal{T}$. Finally, define $\Gamma_i := \partial\Omega_i \setminus \partial\Omega$ and $\Gamma_{ik} = \Gamma_{ki} = \partial\Omega_i \cap \partial\Omega_k$.

3 An Overlapping Schwarz Method

Overlapping Schwarz methods for elliptic problems guarantee good convergence properties if the overlap is sufficiently large. On the other hand, they are not easy to implement. We propose a *modified* Schwarz method for singularly perturbed problems which is more appropriate for parallelization and allows minimal overlap [BS91]. A description in the 2D–case (with obvious modifications in 3D) is as follows: Starting from the non–overlapping partition $\overline{\Omega} = \bigcup_i \overline{\Omega}_i$, we introduce small *interface domains* $\mathcal{O}_{ik}$ covering the interface Γ_{ik} between adjacent subdomains with thickness $\Delta \approx kh$ and *crosspoint regions* $\mathcal{C}$ of diameter kh each covering a crosspoint. Starting from an initial guess $U^0 \in V_0^h(\Omega)$, the iteration method for problem (3) reads for $n \in \mathbf{N}$:

1. Solve in parallel on each subdomain Ω_i:

$$B_{\Omega_i}^{SG}(U_i^{n+1}, v) = L_{\Omega_i}^{SG}(v), \; \forall v \in V_0^h(\Omega_i); \qquad U_i^{n+1} - U_i^n \in V_0^h(\Omega_i).$$

2. Solve in parallel (redundantly) on each interface domain $\mathcal{O} = \mathcal{O}_{ik}$:
$$B_{\mathcal{O}}^{SG}(V_{\mathcal{O}}^{n+1}, v) = L_{\mathcal{O}}^{SG}(v), \ \forall v \in V_0^h(\mathcal{O}); \qquad V_{\mathcal{O}}^{n+1} - U^{n+1} \in V_0^h(\mathcal{O}).$$
Then set $U^{n+1} = V_{\mathcal{O}}^{n+1}$ on the interface Γ_{ik} generating $\mathcal{O}$.
3. Solve in parallel (redundantly) on each crosspoint region $\mathcal{C}$
$$B_{\mathcal{C}}^{SG}(W_{\mathcal{C}}^{n+1}, v) = L_{\mathcal{C}}^{SG}(v), \ \forall v \in V_0^h(\mathcal{C}); \qquad W_{\mathcal{C}}^{n+1} - U^{n+1} \in V_0^h(\mathcal{C}).$$
Then set $U^{n+1} := W_{\mathcal{C}}^{n+1}$ on the corresponding interfaces in the crosspoint region $\mathcal{C}$.
4. Set $n \mapsto n+1$ and goto step 1.

The Schwarz method is similarly defined for the continuous problem (1). We consider the convergence of the (*continuous*) Schwarz method, for simplicity, in the following model problems in $\Omega = (0,1)^2$ with a very simple flow field $\boldsymbol{a}$ and substructuring according to

$$\textbf{(H.1)}^* \qquad \begin{cases} A: & a_1(x) \geq A_1 \gg \epsilon > 0, \quad a_2(x) = 0, \qquad\qquad x \in \Omega \\ B: & a_i(x) \geq A_i \gg \epsilon > 0, \quad i = 1,2 \qquad\qquad x \in \Omega \\ C: & a_i(x) = 0, \quad i = 1,2; \quad c(x) \geq c_0 \gg \epsilon > 0 \quad x \in \Omega, \end{cases}$$

$$\textbf{(H.2)} \qquad \begin{cases} \Omega = (0,1)^2 \text{ is split into non-overlapping subdomains } \Omega_{ij} := \\ ((i-1)H_1, iH_1) \times ((j-1)H_2, jH_2),\ i = 1, ..., M_1,\ j = \\ 1, ..., M_2,\ H_k := \frac{1}{M_k}. \end{cases}$$

The crucial point in (H.1)*, (H.2) is a uniform behaviour of $\mathrm{sgn}(\boldsymbol{a} \cdot \boldsymbol{\nu})$ on the interface between adjacent subdomains. In case B there exist only inflow and outflow parts $\partial\Omega_{ij}^-$ and $\partial\Omega_{ij}^+$. In case A we additionally have characteristic parts $\partial\Omega_{ij}^0$. In case C we obtain the trivial case $\partial\Omega_{ij} = \partial\Omega_{ij}^0$. Assume that the interface regions are generated by narrow strips in $\overline{\Omega}_{ij}$ of thickness Δ_{ij}^-, Δ_{ij}^+ or Δ_{ij}^0 at the inflow outflow or characteristic part of $\partial\Omega_{ij}$. The intersection of the interface strips generates crosspoint regions $\mathcal{C}$.

Theorem 3.1. *Assume* (H.1)*, (H.2) *and set* $TOL > 0$, $K := \|u - u^0\|_{L^\infty(\Omega)}$. *Furthermore assume that the minimal overlap width of the interface strips satisfies*
$$\Delta_{ij}^+, \Delta_{ij}^- \geq \alpha^{-1}\varepsilon \left| \ln \frac{TOL}{4K} \right|, \quad \Delta_{ij}^0 \geq \beta^{-1}\sqrt{\varepsilon} \left| \ln \frac{TOL}{4K} \right|. \tag{4}$$
Then we obtain after $M + k$ *steps of the modified Schwarz method that*
$$\|u - u^{M+k}\|_{L^\infty(\Omega)} \leq C(TOL)^{k+1}, \qquad C \sim M, \tag{5}$$
with $M = M_1$, $M = \sum_{i=1}^2 M_i$ *and* $M = 0$ *in case* A, B *and* C, *respectively, and appropriate* ϵ*–independent constants* $\alpha, \beta > 0$. □

Outline of the proof: The key of the proof is some exponential decay of presumably wrong Dirichlet data (appearing during the iteration) in overlapping regions leading to artificial layers. The proof is based on the barrier function technique using the following variant of the maximum principle on an arbitrary subdomain $G \subset \Omega$ with Lipschitzian and piecewise C^2–boundary: Suppose that for $T, S \in C^2(G) \cap C(\overline{G})$ holds
$$|(L_\varepsilon T)(x)| \leq (L_\varepsilon S)(x), \ x \in G, \ |T(x)| \leq S(x), \ x \in \partial G.$$

Then we obtain $|T(x)| \leq S(x)$ in $\overline{G}$.

Consider now in particular case A: The manifolds ∂G^-, ∂G^+ and ∂G^0 for $G = (a,b) \times (c,d) \subset \Omega = (0,1)^2$ are located at $\{a\} \times (c,d)$, $\{b\} \times (c,d)$ and $[a,b] \times \{c\} \cup [a,b] \times \{d\}$. Let $T_1, T_2 \in C^2(G) \cap C(\overline{G})$ be solutions of (1). Then $T = T_1 - T_2$ satisfies

$$\begin{aligned} |T(x)| \quad &\leq \quad \|T\|_{L^\infty(\partial G^-)} + \exp\left[\frac{-\alpha}{\varepsilon}\mathrm{dist}(x, \partial G^+)\right] \|T\|_{L^\infty(\partial G^+)} \\ &\quad + (1 + x_1 - a) \exp\left[\frac{-\beta}{\sqrt{\varepsilon}}\mathrm{dist}(x, \partial G^0)\right] \|T\|_{L^\infty(\partial G^0)}. \end{aligned}$$

with $0 < \ \alpha \neq \alpha(\varepsilon)$, $0 < \beta \neq \beta(\varepsilon)$. The exponential terms of the barrier function mimic artificial layers of width $0(\epsilon \log 1/\epsilon)$ or $0(\sqrt{\epsilon} \log 1/\epsilon)$ at ∂G^+ and ∂G^0, respectively. They are small in subdomains of G with appropriate distance to ∂G^+ and ∂G^0. A corresponding result holds in case B and C.

The idea in case A and B is a *downwind correction* of the solution from subdomain to subdomain. The chosen thickness of the interface and crosspoint regions guarantees the *exponential decay* of the influence of wrong Dirichlet data in upwind and crosswind directions. The first iteration cycle 1–4 yields in particular in case A an error of $0(TOL)$ in subdomains $\Omega_{1j}, j = 1, ..., M_2$. The next iteration gives an error of $0(TOL)^2$ there and of $0(TOL)$ in subdomains $\Omega_{2j}, j = 1, ..., M_2$. The desired result follows by induction. The idea is the same in case B. In case C we have an *isotropic propagation* of information. After the first iteration cycle the error is of order TOL in all subdomains Ω_{ij}. The result follows then again by induction. □

Remark 3.1. It is possible to extend the result of Theorem 3.1 to 3D and to more general domains and macro partitions, in particular of singularly perturbed diffusion–reaction problems (cf. case C). The case of nonsymmetric singularly perturbed problems is more involved due to the possibly complicated behaviour of the characteristics at the interface $\Sigma := \cup \Gamma_{ik}$.

Furthermore, a result corresponding to Theorem 3.1 can be derived for the energy norm $|||\cdot||| := \sqrt{B^{SG}(\cdot,\cdot)}$. The proof of similar results for the *discrete* Schwarz method depends strictly on the discretization method. Some ideas and technical details for the streamline upwind method which is closely related to (3) can be found in [RZ95], [Zho95]. In particular, a discrete maximum principle is not available. □

4 An Adaptive Non–overlapping Method

Consider again a non–overlapping partition $\overline{\Omega} = \bigcup_i \overline{\Omega}_i$. The results of Sec. 3 indicate that a transition to a non–overlapping method should be possible with appropriate interface conditions at $\Sigma := \cup_{i,k} \Gamma_{ik}$. A first insight is given with the *fictitious overlapping method* [LeT94]. Consider in the (continuous) overlapping method of Sec. 3 with overlap width Δ_{ik} at Γ_{ik} a first order Taylor expansion of the solution at Γ_{ik}. This leads to the non–overlapping Schwarz Method proposed by P. L. Lions [Lio89]. Set $\rho_{ik} = \frac{\epsilon}{\Delta_{ik}}$. Starting from an initial guess u^0, the iterative procedure reads: Solve (in parallel) on Ω_i

$$L_\epsilon u_i^{n+1} = f \quad \text{in } \Omega_i; \qquad u_i^{n+1} = 0 \quad \text{on } \partial\Omega_i \cap \partial\Omega, \tag{6}$$

with interface condition

$$\epsilon \frac{\partial u_i^{n+1}}{\partial \nu_i} + \rho_{ik} u_i^{n+1} = -\epsilon \frac{\partial u_k^n}{\partial \nu_k} + \rho_{ik} u_k^n \text{ on } \Gamma_{ik}. \tag{7}$$

A first convergence result was given by P.L. Lions [Lio89] for the fictitious overlapping method. Assume (H.1) and that no crosspoints occur (strip partitions of Ω):

$$u^n \to u \text{ strongly in } L_2(G),\ G \subset\subset \Omega_i; \quad u^n \to u \text{ weakly in } L_2(\partial\Omega_i) \tag{8}$$

Remark 4.1. A modified approach leading to (6),(7) is the *three–field formulation* of [BBM92] which consists of finding $u = \{u_i\}$ with $u_i \in H^1_{0,\Omega}(\Omega_i)$ and Lagrange multipliers for the Neumann and Dirichlet data on the interface. After using an augmented Lagrangian technique [LeT94] this formulation can be decoupled iteratively ending up with the method (6),(7). □

The result of [Lio89] gives no indication for the *design* of ρ_{ik} in the interface condition (7). Generalizing an idea of [Nat96] we propose the following modification of (7)

$$\rho_{ik} := \rho_i^- := -\frac{1}{2}(\boldsymbol{a} \cdot \boldsymbol{\nu}_i - Z_i|_{\Gamma_{ik}}) = \rho_k^+ := \tfrac{1}{2}(\boldsymbol{a} \cdot \boldsymbol{\nu}_k + Z_k|_{\Gamma_{ik}}) \tag{9}$$

with Z_i a strictly positive function on Γ_i. We analyze the *convergence* of the adaptive method (6),(7), (9) applied to (1).

Theorem 4.1 *Assume* (H.1) *(or* $c - \frac{1}{2}\nabla \cdot \boldsymbol{a} \geq 0$ *and* $\partial\Omega_i \cap \partial\Omega \neq \emptyset\ \forall i$*) and* $\frac{\partial u}{\partial \boldsymbol{\nu}_i}|_{\Gamma_{ik}} \in L^2(\Gamma_{ik})$ *for the solution of (1). Then the sequence* $u^n = \{u_i^n\}$ *generated by algorithm (6),(7),(9) converges for* $n \to \infty$ *with* $u_i^n \to u$ strongly in $H^1(\Omega_i)$.

Outline of the proof: First of all we observe that algorithm (6), (7), (9) is well defined provided that $\frac{\partial u_i^0}{\partial \boldsymbol{\nu}_i} \in L^2(\Gamma_i)$. This implies $\frac{\partial u_i^n}{\partial \boldsymbol{\nu}_i} \in L^2(\Gamma_i)$ for all $n \in \mathbf{N_0}$.
The key step is a modification of Lemma 4.4 in [Nat96]: A function $u \in H^1_{0,\Omega}(\Omega_i)$ with $L_\epsilon u = 0$ in $L^2(\Omega_i)$ and $\frac{\partial u}{\partial \boldsymbol{\nu}_i} \in L^2(\Gamma_i)$ satisfies

$$|||u|||_i^2 + \int_{\Gamma_i} \frac{1}{2Z_i} \left[\left(\epsilon \frac{\partial}{\partial \boldsymbol{\nu}_i} - \rho_i^+ \right) u \right]^2 ds = \int_{\Gamma_i} \frac{1}{2Z_i} \left[\left(\epsilon \frac{\partial}{\partial \boldsymbol{\nu}_i} + \rho_i^- \right) u \right]^2 ds \tag{10}$$

with the outer unit normal ν_i on Γ_i and

$$|||u|||_i^2 := \epsilon \|\nabla u\|^2_{L^2(\Omega_i)} + \|\sqrt{\mu} u\|^2_{L^2(\Omega_i)}, \ \mu := c - \frac{1}{2}\nabla \cdot \boldsymbol{a},\ Z_i \geq z_0 > 0 \text{ a.e. on} \, \Gamma_i.$$

The error $e^{n+1} := u_i - u_i^{n+1}$ satisfies $L_\epsilon e^{n+1} = 0$ in Ω_i, hence

$$|||e_i^{n+1}|||^2_{\Omega_i} + \int_{\Gamma_i} \frac{1}{2Z_i} \left[\left(\epsilon \frac{\partial}{\partial \boldsymbol{\nu}_i} - \rho_i^+ \right) e_i^{n+1} \right]^2 ds = \int_{\Gamma_i} \frac{1}{2Z_i} \left[\left(\epsilon \frac{\partial}{\partial \boldsymbol{\nu}_i} + \rho_i^- \right) e_i^{n+1} \right]^2 ds$$

Using (7),(9), we obtain

$$|||e_i^{n+1}|||^2_{\Omega_i} + \sum_{k(\neq i)} |[e_i^{n+1}]|^2_{\Gamma_{ik}} = \sum_{k(\neq i)} |[e_k^n]|^2_{\Gamma_{ik}}$$

where

$$|[e_i^n]|^2_{\Gamma_{ik}} := \int_{\Gamma_{ik}} \frac{1}{2Z_i} \left[\left(\epsilon \frac{\partial}{\partial \boldsymbol{\nu}_i} - \rho_i^+ \right) e_i^n \right]^2 \, ds.$$

Summing over the subdomains yields

$$\sum_{i=1}^{N} |||e_i^{n+1}|||^2_{\Omega_i} + |[e^{n+1}]|^2_{\Sigma} = |[e^n]|^2_{\Sigma}, \qquad |[e^n]|^2_{\Sigma} := \sum_{i=1}^{N} \sum_{k(\neq i)} |[e_i^n]|^2_{\Gamma_{ik}}$$

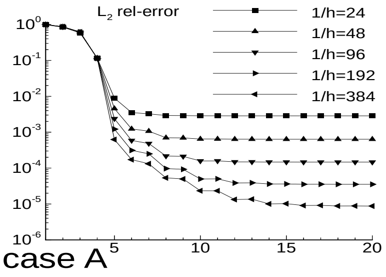
where $\Sigma := \cup_{i=1}^{N} \Gamma_i$. Summation over n implies strong H^1-convergence of $\{e^n\}$ to zero due to the equivalence of norms. □

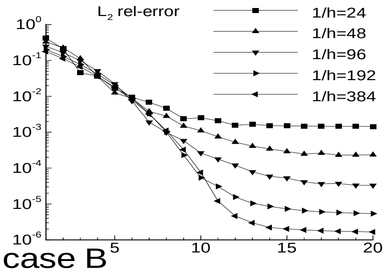
Remark 4.2. Possible choices of Z_i: (i) The choice $Z_i = \sqrt{(\boldsymbol{a} \cdot \boldsymbol{\nu}_i)^2 + 4\epsilon c}$ was derived in [Nat96] from an zeroth order approximate factorization of the elliptic operator L_ϵ. But this choice failed in case $c = 0$ if the flow field is parallel to some Γ_{ik}. In the original proof ([Nat96], Th. 4.1) Robin-type boundary conditions on $\partial\Omega$ were assumed.
(ii) We suggest $Z_i = \sqrt{(\boldsymbol{a} \cdot \boldsymbol{\nu}_i)^2 + \lambda\epsilon}$ with some arbitrary positive parameter λ. Then even if $c = 0$ no restriction to the flow field is necessary. Thus the Theorem is also applicable to the Poisson problem and improves the result of [Lio89].
(iii) With $Z_i =| \boldsymbol{a} \cdot \boldsymbol{\nu}_i |$ one obtains the so-called *adaptive Robin-Neumann algorithm* [CQ95], [Tro96], [GGQ]. Then Theorem 4.1 is applicable only if $| \boldsymbol{a} \cdot \boldsymbol{\nu}_i |> 0$. In case of two subdomains [GGQ] prove weak H^1-convergence under the more general assumption that $| \boldsymbol{a} \cdot \boldsymbol{\nu}_i |$ can vanish in a finite number of points. □
The result of Theorem 4.1 means that the corresponding Poincare–Steklov operator is strictly non–expansive and gives no information on the convergence rate.
On the other hand, with our choice for Z_i very reasonable results are obtained for a *discrete version* of the adaptive non–overlapping method: Instead of (6),(7),(9) one has to solve for $n \in \mathbf{N}_0$ (in parallel for $i = 1, ..., I$)

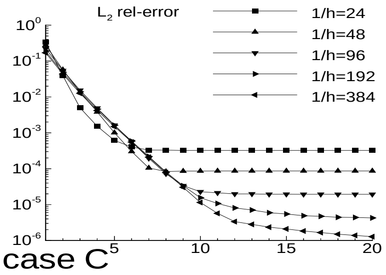
$$B^{SG}_{\Omega_i}(U_i^{n+1}, v) + \sum_{k(\neq i)} \left(\rho_i^+ U_i^{n+1} - \Lambda^n_{ik}, v \right)_{\Gamma_{ik}} = L^{SG}_{\Omega_i}(v), \quad \forall v \in V^h_{\Omega_i}, \qquad (11)$$

$$\Lambda^{n+1}_{ik} := \left(\rho_i^+ + \rho_i^- \right) U_k^{n+1} - \Lambda^n_{ki} = Z_{ik} U_k^{n+1} - \Lambda^n_{ki}. \qquad (12)$$

Here we present some *numerical results* for 2D–problems satisfying (H.1)*, (H.2).
Examples: We choose $\epsilon = 10^{-4}$, a $M_1 \times M_2$-partition of Ω and a sequence of different values of h. The (continuous) solution is always $w = \sin \pi x_1 \left(\exp \left(x_2^2 - x_2 \right) - 1 \right)$, hence $f = L_\epsilon w$. More precisely, we consider the advection dominated case with (B) and without (A) characteristic interfaces and the reaction dominated case (C) (cf. also Th. 3.1 for the weakly overlapping method).







Here we show the convergence history of the relative discrete L^2-norm vs. the iteration number for the test cases A, B and C for different values of h. For the parameter λ in our formula for Z_i we have chosen 100, 100 and 10 for case A,B and C respectively.

Case A: $\boldsymbol{a} = \frac{1}{\sqrt{5}}(2,1)^T$, $c = 0$ and $M_1 = M_2 = 3$
Case B: $\boldsymbol{a} = (0,1)^T$, $c = 0$ and $M_1 = 4$, $M_2 = 1$
Case C: $\boldsymbol{a} = (0,0)^T$, $c = 1$ and $M_1 = M_2 = 3$

The convergence history is similarly as predicted by Theorem 3.1 for the weakly overlapping Schwarz method: In case A, we observe an initial phase of downwind propagation of information, then we obtain reasonable linear convergence until the discretization error level is reached. In cases B and C, no initial advective propagation of information appears. Reasonable linear convergence is again obtained. The algorithm is only slightly sensitive w. r. t. λ.
A heuristic explanation of the favourable convergence properties of the discrete algorithm can be given by means of singular perturbation arguments. In cases A and C, wrong interface data cause artificial layers of exponential (and essentially 1D-) type for the flux $\epsilon \nabla u \cdot \nu$. The analysis is more involved in case B. Wrong interface data cause artificial layers which are of parabolic type for the flux. A higher order factorization of the operator L_ϵ would better represent the advective transport along the interface [Nat96]. Fortunately, the diffusive interface transport involved in the transmission condition (7), (9) is obviously sufficient to guarantee convergence. Nevertheless, further theoretical foundation of the convergence properties of the discrete algorithm (6), (7) is neccesary.

5 Summary and Open Problems

Two parallelizable methods are considered for singularly perturbed elliptic problems. We obtain linear convergence for the overlapping method in the continuous case. The overlap width can be minimized for advection and/ or reaction dominated problems. The non–overlapping method with properly problem adapted interface condition of Robin type gives strong H^1-convergence in the continuous case. The method applied to the discrete problem provides reasonable performance in the range from diffusion to advection (or reaction) dominated problems. Nevertheless, there are still open problems concerning the convergence analysis even for scalar elliptic problems. The application to incompressible flow problems is considered in a forthcoming paper.

Acknowledgement

The support by BMBF under grant No. 03LU7GOE and the University of Magdeburg, IAN is gratefully acknowledged.

REFERENCES

[BBM92] Baiocchi C., Brezzi F., and Marini L. (1992) *Stabilization of Galerkin methods and applications to domain decomposition.* In Bensoussan . A. and Verjus J.-P. (eds) *Future Tendencies in Computer Science, Control and Applied Mathematics.* Springer-Verlag, Berlin Heidelberg.

[BS91] Boglaev I. and Sirotkin V. (1991) *Domain decomposition technique for singularly perturbed problems.* In Miller . J. and Vichnevetsky R. (eds) *Proc. 13. IMACS.* Dublin.

[CQ95] Carlenzoli C. and Quarteroni A. (1995) *Adaptive domain decompositon methods for advection-diffusion problems.* In Babuška . I. *et al.* (eds) *Modeling, Mesh Generation, and Adaptive Numerical Method for Partial Differential Equations,* Institute for Mathematics and its Applications IMA Volume 75. Springer Verlag, New York a.o.

[FFLR96] Franca L., Farhat C., Lesoinne M., and Russo A. (1996) *Unusual stabilized finite element methods and residual-free-bubbles.* Technical report, UCD/ UMD Report No. 82. University of Colorado.

[GGQ] Gastaldi F., Gastaldi L., and Quarteroni A.*Adaptive domain decomposition methods for advection dominated equations.* to appear in: East West J. Num. Math.

[LeT94] LeTallec P. (1994) *Domain decomposition methods in computational mechanics.* Comput. Mech. Adv. **1** pages 121–220.

[Lio90] Lions P. (1990) *On the Schwarz Alternating Method III: A Variant for Nonoverlapping Subdomains.* In Chan . T., Glowinski R., Périaux J., and Widlund O. (eds) *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations.* SIAM, Philadelphia.

[NR95] Nataf F. and Rogier F. (1995) *Factorization of the convection-diffusion operator and the schwarz algorithm.* Math. Model. Meths. Appl. Sc. **5** pages 67–93.

[RZ95] Rannacher R. and Zhou G. (1995) *Analysis of a domain-splitting method for nonstationary convection–diffusion problems.* East-West J. Numer. Math. **2** pages 151–172.

[Tro] Trotta R.*Multidomain finite elements for advection–diffusion equations.* to appear in: Appl. Numer. Math.

[Zho95] Zhou G. (1995) *A new domain decomposition method for convection–dominated problems.* Technical report, Preprint 95–38 (SFB 359). University of Heidelberg.

20

Convergence Rate of Schwarz-type Methods for an Arbitrary Number of Subdomains

Frederic Nataf and Francis Nier

1 Introduction

The original Schwarz method is based on the use of Dirichlet boundary conditions as interface conditions (see [Lio89] and references therein). Convergence can be reached only with overlapping subdomains. As a result, the convergence is very slow when the overlap is small. In order to speed up the convergence and to be able to handle nonoverlapping subdomains, it has been proposed in [Lio89] to replace the Dirichlet boundary conditions by Fourier or more complex boundary conditions. The question is then to choose the best interface conditions both in terms of convergence rate and of easiness of use and implementation. In order to make a proper choice, it is important to quantify the effect of the boundary conditions on the convergence. This is usually done by a Fourier analysis for a two-domain decomposition (see [Des90], [CNR91], [CQ93], [TB94], [NR95], and [Jap96]).

A natural question is then what happens when there are more than two subdomains. In [NN97], we prove the convergence of three Schwarz-type algorithms. We establish a link between the convergence rate of a two-subdomain decomposition and the convergence for a decomposition into an arbitrary number of subdomains.

2 Two-domain Decomposition Convergence Rate

In order to find good interface conditions, a common practice is to consider the problem set on $\mathbf{R}^2$ decomposed into two half-planes. If we have to solve $\mathcal{L}(u) = f$ on $\mathbf{R}^2$, the Schwarz algorithm is

$$\mathcal{L}(u_1^{n+1}) = f \text{ in } \mathbf{R}^2_-, \quad \mathcal{B}_{12}(u_1^{n+1}) = \mathcal{B}_{12}(u_2^n) \text{ at } x = 0$$
$$\mathcal{L}(u_2^{n+1}) = f \text{ in } \mathbf{R}^2_+, \quad \mathcal{B}_{21}(u_2^{n+1}) = \mathcal{B}_{21}(u_1^n) \text{ at } x = 0$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

The operators $\mathcal{B}_{12}$ and $\mathcal{B}_{21}$ are interface conditions to be chosen. As an example, $\mathcal{B}_{12,21}$ can be sought in the form of a partial differential operator of order two in the tangential direction $\mathcal{B}_{12} = \partial_x + \alpha + \beta\partial_y + \gamma\partial^2_{yy}$. The convergence rate will depend on the value of the coefficients α, β and γ. Usually the study is made by freezing the coefficients so that Fourier transform in the direction y can be used. The dual variable is denoted by k. It is then easy to compute explicitly a formula for the convergence rate ρ as a function of the Fourier variable k. For instance, if absorbing boundary conditions of order 0 wrt k are used for $\mathcal{B}_{12,21}$, ρ equals zero at $k = 0$ and tends to 1 as k tends to infinity (see [Des90], [NR95]). One may also be tempted to optimize the convergence rate with respect to some of the coefficients α, β or γ (see [TB94], [Jap96]). This kind of study yields valuable information. For more details on the importance of interface conditions and also a numerical study, see the proceedings of C. Japhet [Jap96] in this volume.

Now, a natural question is what happens when there is an arbitrary number N of subdomains. The answer is not obvious since it amounts to estimating the norms of a $2N - 2$ squared matrix raised to any power n.

3 The Main Result

In [NN97], we prove the convergence of three Schwarz-type methods with or without overlapping. We also establish a link between the two-domain convergence rate and the convergence for an arbitrary number of subdomains.

The space $\mathbf{R}^{d+1}$ is decomposed into N vertical strips. A constant coefficient advection-diffusion equation is solved: $\mathcal{L}(u) = f$. The velocity in the direction x is positive. The flow goes from the left to the right. Three methods are considered. In the first method, the update is simultaneous in all the subdomains. This is the additive Schwarz method (ASM). In the second algorithm, the update is made sequentially by sweeping over the domains following the direction of the flow (FDS) (see [HIKW92], [Joh92], [Nat96]). The last method is a variation of the previous one. The approximations in the subdomains are updated by double sweeps over the subdomains (DS).

The result is the following

Theorem 3.1 *There exists a function $\rho(k)$ taking its values in $[0, 1)$ independent of N so that $\hat{e}^n_i(k)$ the k-th component of the error in the Fourier space for the method i is estimated as follows:*

$$\|\hat{e}^n_i(k)\| \leq C_i(k)\rho(k)^{[n/p_i]} \textit{ for } n \geq n_i,$$

where $[m]$ denotes the integer part of m. The values of p_i and n_i depend on the method:

$$p_{ASM} = 2N-2,\ p_{FDS} = N-1,\ p_{DS} = 1,\ n_{ASM} = 2N+1,\ n_{FDS} = 2N-1,\ n_{DS} = 3.$$

Due to the ellipticity of the operator, we also prove that the function ρ is almost equal to the two-domain decomposition convergence rate. A connection is thus established between the study of convergence with two subdomains and the case with N subdomains.

The proof is unusual since it relies on techniques originating in formal language theory. It is worth noticing that this result is sharper than the estimate of the spectral radius of the iteration matrix. Indeed, the result would be something like: for any positive ϵ there exists some positive constant C_ϵ so that for n larger than some integer n_ϵ, the error is bounded by $C_\epsilon\,(\rho+\epsilon)^n$. While, here, the constant is known explicitly and the estimate is valid from a rank which is an explicit function of N.

4 Sketch of the Proof

Reformulation of the Algorithm

The proof is in two steps. First the algorithms are reformulated so that the unknowns are functions living on the boundaries of the subdomains. Then, we conclude with an algebraic trick.

The first part is very classical (see [NRdS94], [NN97]). Let H denote a vector of functions living on the boundaries of the subdomains. It may be seen that H satisfies a linear equation:

$$(Id-\mathcal{T})(H)=G$$

The matrix-vector product $\mathcal{T}(H)$ consists in solving in parallel a boundary value problem in each subdomain. The operator $\mathcal{T}$ is split into the sum of four operators $\mathcal{T}=\mathcal{T}_{ll}+\mathcal{T}_{rr}+\mathcal{T}_{rl}+\mathcal{T}_{lr}$. This enables us to give a compact form of the iteration matrices of the different algorithms (simply $\mathcal{T}$ for the additive Schwarz method, $\mathcal{T}_{FDS}$ for the flow directed algorithm and by $\mathcal{T}_{DS}$ for the double sweeps algorithm):

$$\begin{aligned}\mathcal{T}_{FDS}&=(Id-\mathcal{T}_{ll}-\mathcal{T}_{lr})^{-1}(\mathcal{T}_{rl}+\mathcal{T}_{rr})\\ \mathcal{T}_{DS}&=(Id-\mathcal{T}_{rr}-\mathcal{T}_{rl})^{-1}(Id-\mathcal{T}_{ll}-\mathcal{T}_{lr})^{-1}(\mathcal{T}_{ll}+\mathcal{T}_{lr})(\mathcal{T}_{rl}+\mathcal{T}_{rr})\end{aligned}$$

The basis for the algebraic trick is the following set of relations:

$$\begin{array}{lll}\mathcal{T}_{ll}^{N-1}=\mathcal{T}_{rr}^{N-1}=0; & \mathcal{T}_{ll}\,\mathcal{T}_{rr}=\mathcal{T}_{rr}\,\mathcal{T}_{ll}=0; & \mathcal{T}_{rl}^{2}=\mathcal{T}_{lr}^{2}=0\\ \mathcal{T}_{lr}\,\mathcal{T}_{ll}=\mathcal{T}_{rl}\,\mathcal{T}_{rr}=0; & \mathcal{T}_{ll}\,\mathcal{T}_{rl}=\mathcal{T}_{rr}\,\mathcal{T}_{lr}=0 & \end{array}\qquad(1)$$

It is worth noticing that these relations come from the structure of the matrices and do not depend on the value of the coefficients.

By using formal language theory, we prove the following.

Theorem 4.1 *If*

$$\rho=\|\mathcal{T}_{rl}\|\,\|\mathcal{T}_{lr}\|\,(\sum_{i=0}^{N-2}\|\mathcal{T}_{rr}^{i}\|)\,(\sum_{i=0}^{N-2}\|\mathcal{T}_{ll}^{i}\|)<1,$$

Then,

$$\begin{aligned}\|\mathcal{T}^n\|&\le C(1-\rho)^{-1}\,\rho^{[n/(2N-2)-3/2]}\quad for\ \ n\ge 2N\\ \|\mathcal{T}_{FDS}^n\|&\le C(1-\rho)^{-1}\,\rho^{[n/(N-1)-2]}\quad for\ \ n\ge 2N-2\\ \|\mathcal{T}_{DS}^n\|&\le C(1+\rho)\,\rho^{n-1}\quad for\ \ n\ge 2\end{aligned}$$

where the constant C is given explicitly

$$C = (1 + \rho/\|\mathcal{T}_{lr}\|)\,(1 + \rho/\|\mathcal{T}_{rl}\| + \rho/\|\mathcal{T}_{rl}\|\|\mathcal{T}_{lr}\|)$$

Another way to look at this result is to remark that when $\mathcal{T}_{lr}$ or $\mathcal{T}_{rl}$ is zero, the operators $\mathcal{T}$, $\mathcal{T}_{FDS}$ and $\mathcal{T}_{DS}$ are nilpotent at different orders. When $\mathcal{T}_{lr}$ and $\mathcal{T}_{rl}$ are not zero, this result quantifies the perturbation to nilpotency they bring. Les us emphasize the fact that the proof is purely combinatorial. We never use the explicit form of the operators $\mathcal{T}$.

5 Complete Proof of a Simplified Result

In this section, we give a flavor of the proof of the algebraic result. We prove the simplest statement related to our techniques by way of example. (We remark that this particular statement could be obtained in other ways.)

Theorem 5.1 *Let T and A be two operators so that T is nilpotent of order $N-1$ and $\rho = \|A\| \sum_{i=0}^{N-2} \|T^i\| < 1$. Then, we have the following estimate:*

$$\|(T+A)^i\| \leq C\rho^{[i/(N-1)]}$$

where C is given explicitly.

A simple estimate

A simple way (in our context) to look at this problem is to use the standard estimate:

$$\|(T+A)^i\| \leq \|(T+A)\|^i \leq (\|T\| + \|A\|)^i \tag{2}$$

But, in our case, this estimate is very poor since it does not use the nilpotency of T. Indeed, in the expansion of $(T+A)^i$,

$$(T+A)^i = T^i + AT^{i-1} + TAT^{i-2} + T^2AT^{i-3} + \dots$$

many terms are zero. More precisely, all the terms containing T^{N-1} are zero. There are many terms of this kind. The problem is how to track them so that to improve (2). It is at this point that formal language theory is relevant. It will enable us to track rigorously the terms which are known to be zero.

Elements of Formal Language Theory

Let us introduce some definitions and concepts dealing with formal language theory. Let t and a be two letters. By combining these letters, it is possible to form *words* e.g. at, att (also denoted at^2), and so on. These words can also be combined to form words by concatenation; e.g. $at.t^3 = at^4$. We also say that at^4 is the product of at by t^3. It is very convenient to introduce a neutral word 1 for this operation: $1.w = w$ for any word w. For a word m we define its length $|m|$ as the number of letters it is made of, e.g. $|at^3a| = 5$.

Another important concept is the lexis X^* generated by a set of words X: it is the set of words obtained by concatenations of the words of X plus the neutral word 1, e.g.

$$X = \{t^2, at\}, \quad X^* = \{1,\, at,\, t^2,\, t^2at,\, at^3,\, atat,\, t^4, \dots\}$$

The generating set X is said to be free if for any word in X^* there is only way to write it as a product of words of X. In the previous example, X is free. This is not always the case as is shown in the next example:

$$X_1 = \{t^4,\, t^6\},\ t^{12} \in X_1^* \text{ and } t^{12} = t^6.t^6 = t^4.t^4.t^4\ .$$

When a basis X is free, we can define without ambiguity the length of a word m in X^* relatively to X. It is the number of words of X m is made of and it is denoted by $|m|_X$, e.g. with X defined as above, $|at^3|_X = 2$ (while $|at^3| = 4$).

Our problem deals with norms of operators and not with words. We need a bridge between normed operators and words. It is the operator *mop* from the lexis $\{t, a\}^*$ to the set of matrices. To a word, we associate the corresponding product of matrices, e.g. $mop(t) = T$, $mop(a) = A$, $mop(atta) = AT^2A$. We also define the weight of a word m as $\|m\| = \|mop(m)\|$ and the weight of a set of words P by $\|P\| = \sum_{m\in P} \|m\|$. The following inequalities will be useful in the sequel.

Property 1 Let P be a free generating set of weight smaller than one; then, we have

$$\|\{m \in P^*/\|m\|_P \geq j\}\| \leq \frac{\|P\|^j}{1 - \|P\|}$$

Property 2 Let P_1, P_2 be two sets of words; we have:

$$\|P_1.P_2\| \leq \|P_1\|.\|P_2\|$$

where $P_1.P_2$ is the set of all the products of a word of P_1 by a word of P_2.

The proofs are very simple and may be found in [NN97].

Proof

At this point, we have all that is necessary in order to prove the theorem.

In our context, it is interesting to look at

$$W = \{m \in \{a, t\}^*/t^{N-1} \text{ is not a substring of } m\}.$$

Indeed, we know that for every word m not in W, the corresponding operator $mop(m)$ is zero. It can be seen easily that W can be factorized as

$$W = \{1, t, t^2, \dots, t^{N-2}\}.\{a, at, \dots, at^{N-2}\}^* \tag{3}$$

Let $W_1 = \{1, t, t^2, \dots, t^{N-2}\}$.

We have to estimate the norm of $(A + T)^j$. The first equality will be obtained by writing $(A + T)^j$ as the sum of $mop(m)$ over the words m of length j. By noticing that for a word m not in W, $mop(m) = 0$, we see that the sum can be taken over W.

$$\|(A+T)^j\| = \|\sum_{|m|=j,\, m\in\{a,t\}^*} mop(m)\| = \|\sum_{|m|=j,\, m\in W} mop(m)\|.$$

Now, by factorization (3) of W, we have that a word m in W can be written as a product $w_1 m_P$ with $w_1 \in W_1$ and $m_P \in P^*$. Since the length of a word in W_1 is smaller or equal to $N-2$, the length of m_P is larger or equal to $j-(N-2)$. By using this and property 2, we get the estimate:

$$\|(A+T)^j\| \leq \|W_1\| \; \|\{m \in P^*/|m| \geq j-(N-2)\}\|.$$

In order to continue, we remark that the larger length of a word in P is $N-1$. It follows that a word in P^* whose length is larger than $j-(N-2)$ has a length relative to P larger than $\frac{j-(N-2)}{N-1}$. Hence,

$$\|(A+T)^j\| \leq \|W_1\| \; \|\{m \in P^*/|m|_P \geq \frac{j-(N-2)}{N-1}\}\|.$$

In order to conclude the proof, we simply apply Property 1 and obtain

$$\|(A+T)^j\| \leq \frac{\|W_1\|}{1-\|P\|} \; \|P\|^{[\frac{j-(N-2)}{N-1}]}.$$

with $\|P\| \leq \rho$.

6 Conclusion

We have given a unified proof of convergence for three Schwarz-type algorithms. We have also established a link between the convergence rate for a two-domain decomposition and for a decomposition into an arbitrary number of subdomains.

We see at least two continuations to this work. First, it would be interesting to extend our proof to the case of an arbitrary decomposition. Second, we have studied Schwarz-type methods which can be seen as Jacobi or Gauss-Seidel algorithms applied to the substructured problem. It is possible that formal language theory could also be applied successfully to general iterative methods such as GMRES or BICGSTAB algorithms when applied to problems of this type:

$$(Id-(T+A))(H) = G$$

where T is nilpotent.

REFERENCES

[CNR91] Charton P., Nataf F., and Rogier F. (1991) Méthode de décomposition de domaine pour l'équation d'advection-diffusion. *C.R. Acad. Sci.* pages 623–626. t. 313, Série I, Paris.

[CQ93] Carlenzoli C. and Quarteroni A. (1993) *Adaptive Domain Decomposition Methods for Advection-Diffusion Problems.* Proceedings of the IMA Workshop on Mesh Adaptivity.

[Des90] Despres B. (1990) *Décomposition de domaine et problème de Helmholtz*, volume 311 of *C.R. Acad. Sci., Paris*, pages 313–316. Série I.

[HIKW92] Han H., Il'in V., Kellogg R., and Wei Y. (1992) Analysis of flow directed iterations. *J. Comput. Math.* 10: 57–76.

[Jap96] Japhet C. (1996) *Optimisation of interface conditions in DDM. Application to convection-diffusion problems.* in this volume.

[Joh92] Johnson C. (1992) *Flow Directed Gauss-Seidel Iterative Methods for Stationary Convection-Diffusion Problems.* Preprint Dept. of Math. Chalmers Univ. of Technology, 1992:29.

[Lio89] Lions P. L. (1989) *On the Schwarz Alternating Method III: A variant for Nonoverlapping Subdomains*, pages 202–223. Third International Symposium on Domain Decomposition Methods for Partial Differential Equations, SIAM.

[Nat96] Nataf F. (1996) Absorbing boundary conditions in block gauss-seidel methods for convection problems. *Mathematical Models and Methods in Applied Sciences* 6: 481–502.

[NN97] Nataf F. and Nier F. (1997) Convergence rate of some domain decomposition methods for overlapping and nonoverlapping subdomains. *Numerische Mathematik* 75: 357–377.

[NR95] Nataf F. and Rogier F. (1995) Factorisation of the convection-diffusion operator and the schwarz algorithm. *Mathematical Models and Methods in Applied Sciences* 5: 67–93.

[NRdS94] Nataf F., Rogier F., and de Sturler E. (1994) *Optimal interface conditions for domain decomposition methods.* Rapport interne n^0 301, CMAP, Ecole Polytechnique.

[TB94] Tan K. H. and Borsboom M. J. A. (1994) *On Generalized Schwarz Coupling Applied to Advection-Dominated Problems*, volume 180 of *Contemporary Mathematics*, pages 125–130.

21

Multilevel Finite Element Riesz Bases in Sobolev Spaces

Rudolf Lorenz and Peter Oswald

1 Introduction

In this note we discuss some results concerning multilevel finite element schemes of hierarchical basis (HB) type in connection with discretizing and preconditioning elliptic problems in Sobolev spaces. Roughly speaking, HB-methods require the introduction of a hierarchically defined algebraic basis Ψ_j of locally supported functions for a scale of finite element discretization spaces V_j, $j \geq 0$, and aim at reducing the condition number of discretization matrices for standard elliptic problems when represented in the basis Ψ_j. Motivated by recently proposed modifications to the standard HB-method (Yserentant [Yse86]) such as the 3-point HB-method of Stevenson [Ste96, Ste97a], the coarse–grid stabilized HB-methods of Carnicer/Dahmen/Peña [CDP96], Vassilevski/Wang [VW97a, VW97b] and the L_2-semiorthogonal prewavelet methods (see [Osw94, Jun94, KO96, Ste97b]), we started in [LO96] a systematic comparison of their properties. In a first step, we considered finite element HB-methods with respect to shift-invariant, dyadically refined triangulations of $\mathbb{R}^d$, and studied the range of the smoothness parameter s for which a given HB-system $\Psi = \cup_{j\geq 0}\Psi_j$ is a Riesz basis in $H^s(\mathbb{R}^d)$. For those s, discretizations of H^s-elliptic problems in V_j with respect to Ψ_j will lead to stiffness matrices with uniformly (j-independent) condition numbers, thus resulting in an asymptotically optimal preconditioning method.

We concentrate here on the case of linear finite elements and $d \leq 3$. Section 2 contains the definitions of HB-systems and a brief survey of the connection to multilevel preconditioners. In Section 3, we report on results obtained in [LO96, LO97a] for the shift-invariant case. Future research should include extensions of the theory to realistic domains and partition sequences obtained by adaptive refinement, as well as a more quantitative investigation of work estimates (condition numbers versus arithmetical complexity per iteration). In Section 4, we provide the condition numbers for generic H_0^1– and L_2–discretizations on a square in $\mathbb{R}^2$ when using the HB-examples discussed in Section 3.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 HB-Systems for Linear Finite Elements

Throughout this paper, let

$$V_0 \subset V_1 \subset \ldots \subset V_j \equiv S_1^0(\mathcal{T}_j) \cap L_2(\Omega) \subset \ldots \tag{1}$$

be the sequence of linear finite element spaces with respect to uniformly and dyadically refined simplicial partitions $\mathcal{T}_j$ of element size $\approx 2^{-j}$ of a polyhedral domain Ω. Specifically, as a model case, we consider $\Omega = \mathbb{R}^d$ and the sequence of shift-invariant (2^d-1)-directional partitions $\mathcal{T}_j$. The nodal basis (NB) functions for V_j will be denoted by $\phi_{j,P}$, $P \in \mathcal{V}_j$, where $\mathcal{V}_j$ is the vertex set of $\mathcal{T}_j$. We set $\mathcal{W}_j = \mathcal{V}_j \backslash \mathcal{V}_{j-1}$ for the sets of vertices newly generated when refining $\mathcal{T}_{j-1}$, $j \geq 1$, $\mathcal{W}_0 = \mathcal{V}_0$. Points in $\mathcal{W}_j$ are the edge midpoints of $\mathcal{T}_{j-1}$. Finally, let $n_j = \#\mathcal{V}_j$, $m_j = \#\mathcal{W}_j$.

The HB-systems we look for are of the form

$$\Psi = \bigcup_{j=0}^{\infty} \{\psi_{j,P} \,:\, P \in \mathcal{W}_j\} \,, \quad \Psi_J = \bigcup_{j=0}^{J} \{\psi_{j,P} \,:\, P \in \mathcal{W}_j\} \,, \tag{2}$$

where the locally supported HB-functions

$$\psi_{j,P} = \sum_{Q \in \mathcal{V}_j} a_{j;P,Q} \phi_{j,Q} \,, \quad P \in \mathcal{W}_j \,, \tag{3}$$

are given by their masks $(a_{j,P,\cdot})$. We assume that the size of these masks (i.e., the number of nonzero coefficients in (3)) is uniformly bounded with respect to j and P. This implies that the rectangular matrices

$$\hat{I}_j = ((a_{j;P,Q}))_{Q \in \mathcal{V}_j, P \in \mathcal{W}_j} \tag{4}$$

of dimension $n_j \times m_j$ have $\mathrm{O}(m_j)$ non-zero entries. We assume that the system of level-j HB-functions $\{\psi_{j,P} \,:\, P \in \mathcal{W}_j\}$ forms an L_2-stable basis in its L_2-closed span W_j, and that V_j admits an L_2-stable direct sum decomposition $V_j = V_{j-1} \dot{+} W_j$. Here, L_2-stability means that

$$\| \sum_{P \in \mathcal{W}_j} c_{j,P} \psi_{j,P} \|_{L_2}^2 \asymp 2^{-jd} \sum_{P \in \mathcal{W}_j} c_{j,P}^2 \tag{5}$$

for all reasonable coefficient choices resp.

$$\|v_{j-1} + w_j\|_{L_2}^2 \asymp \|v_{j-1}\|_{L_2}^2 + \|w_j\|_{L_2}^2 \qquad \forall\, v_{j-1} \in V_{j-1} \,,\ \forall\, w_j \in W_j \,. \tag{6}$$

We always assume that two-sided estimates expressed by $\asymp$ hold with positive constants that are independent of parameters and functions, especially, of j. The assumptions (5), (6) are usually easy to check (since they concern only two adjacent levels), and imply that the finite sections Ψ_J of the HB-system Ψ are algebraic bases in V_J, for all $J \geq 0$.

However, there is no guarantee for uniform L_2-stability of the Ψ_J or for stability of the whole HB-system Ψ in the L_2-norm (or in other norms) under the above assumptions. This desirable property is, up to scaling, part of the definition of a Riesz basis.

Definition 1 *A system $\mathcal{F} \equiv \{f_l\} \in H$ is a Riesz basis in the (real) Hilbert space H if the mapping*

$$(c_l) \longmapsto \sum_l c_l f_l \ ,$$

which is well–defined for finite sequences (c_l), can be extended to an isomorphism between l_2 and H. In other words, $\mathcal{F}$ should be dense and minimal in H, and satisfy

$$\| \sum_l c_l f_l \|_H^2 \asymp \sum_l c_l^2 \ .$$

The best possible constants in this two-sided inequality are called Riesz bounds of $\mathcal{F}$ in H.

For properties of Riesz bases and frames (the latter generalize the stabilty concept to nonunique decompositions and generating systems) in connection with multiresolution analysis and multilevel methods, see [Dau92, Dah96, Osw97]. We quote a corollary for the finite element HB-systems introduced above when applied to variational problems in Sobolev spaces $H^s(\Omega)$. Consider the symmetric $H^s(\Omega)$-elliptic variational problem of determining $u \in H^s(\Omega)$ such that

$$a(u,v) = \langle f,v \rangle_{H^{-s}\times H^s} \qquad \forall\, v \in H^s(\Omega) \ . \tag{7}$$

We can restrict (7) to V_J: Find $u_J \in V_J$ such that

$$a(u_J,v_J) = \langle f,v_J \rangle_{H^{-s}\times H^s} \qquad \forall\, v_J \in V_J \ . \tag{8}$$

Naturally, for C^0 finite elements, $s < 3/2$ has to be assumed. For finite-dimensional V_J, (8) leads to different linear systems depending on the choice of a basis in V_J. The choice $\{\phi_{J,P} \ : \ P \in \mathcal{V}_J\}$ leads to the standard NB discretization

$$A_J x_J = f_J \ , \tag{9}$$

with $a(\phi_{J,P},\phi_{J,Q})$ resp. $\langle f,\phi_{J,P}\rangle_{H^{-s}\times H^s}$ being the entries of the matrix resp. right-hand side of (9). Analogously, taking Ψ_J, we get

$$A_J^\Psi y_J = f_J^\Psi \ . \tag{10}$$

The solution vectors $x_J = (x_{J,P} : \ P \in \mathcal{V}_J)$ and $y_J = (y_{j,P} : \ P \in \mathcal{W}_j, j \le J)$ represent the NB and HB coefficients of the solution u_J of (8), i.e.,

$$u_J = \sum_{P\in\mathcal{V}_J} x_{J,P}\phi_{J,P} = \sum_{j=0}^{J} \sum_{P\in\mathcal{W}_j} y_{j,P}\psi_{j,P} \ .$$

If we denote the matrix for the change of basis between HB- and NB-representations of functions from V_J by S_J^Ψ (e.g., $x_J = S_J^\Psi y_J$) then one easily sees that

$$A_J^\Psi = (S_J^\Psi)^T A_J S_J^\Psi \ . \tag{11}$$

Note that due to (3) a multiplication by S_J^Ψ can be implemented in $\asymp n_J$ operations. The constants depend on the mask size bound.

As is well-known, the condition numbers of A_J exhibit exponential growth $\asymp 2^{2|s|J}$ for $s \neq 0$. A desirable feature of a HB-construction would be to get J-independent, uniformly bounded condition numbers for the HB-stiffness matrix A_J^Ψ. Using the sparse S_J^Ψ transformation, this would immediately lead to economic iterative solvers for (8). The theoretical answer is

Theorem 1 *Suppose* $\dim V_j < \infty$ *and* $s < 3/2$. *Then, the following are equivalent:*

(i) The normalized HB-system

$$\tilde{\Psi} = \bigcup_{j=0}^{\infty} \{ \|\psi_{j,P}\|_{H^s(\Omega)}^{-1} \psi_{j,P} \ : \ P \in \mathcal{W}_j \}$$

is a Riesz basis in $H^s(\Omega)$.

(ii) The HB-discretization matrices A_J^Ψ *in (10) associated with a symmetric* $H^s(\Omega)$*-elliptic variational problem (7) possess uniformly bounded condition numbers after diagonal scaling.*

The upper bound for $\kappa((D_J^\Psi)^{-1} A_J^\Psi)$, *where* D_J^Ψ *is the diagonal part of* A_J^Ψ, *depends on the Riesz bounds of* $\tilde{\Psi}$ *and the ellipticity constants of the form* $a(\cdot,\cdot)$ *(i.e., on the constants in* $a(u,u) \approx \|u\|_{H^s}^2$, $u \in H^s(\Omega)$*).*

We omit the proof which can be given by using the theory of stable subspace splittings [Osw94], compare also [Osw97]. A reformulation of Theorem 1 (ii) is that

$$\kappa(C_J^\Psi A_J) = \mathrm{O}(1) , \quad J \to \infty , \quad C_J^\Psi = S_J^\Psi (D_J^\Psi)^{-1} (S_J^\Psi)^T . \tag{12}$$

The product $C_J^\Psi A_J$ coincides with the matrix representation of the additive Schwarz operator associated with the splitting

$$V_J = \sum_{j=0}^{J} \sum_{P \in \mathcal{W}_j} W_{j,P} \qquad (W_{j,P} = \operatorname{span} \psi_{j,P})$$

of V_J into the direct sum of one-dimensional subspaces $W_{j,P}$ each of which is spanned by a single HB-function of some level $j \leq J$. The scalar products are induced by $a(\cdot,\cdot)$. See [Osw94, Osw97, LO96, LO97a] for more details, also on the recursive definition of the symmetric preconditioner C_J^Ψ which, besides the diagonal scaling, involves the matrices $\hat{I}_j$ (which actually describe the embedding $W_j \subset V_j$), and analogous matrices I_j describing the embedding $V_{j-1} \subset V_j$, $j = 1, \ldots, J$.

3 Riesz Bases in $H^s(\mathbb{R})^d$: Examples

In general, the verification of the Riesz basis property of a given HB-system in Sobolev spaces is not trivial. It has to do with tools like Jackson-Bernstein inequalities for scales of approximating spaces (such as $\{V_j\}$) but also with the study of associated biorthogonal systems. We refer to [Dah96]. A considerable simplification is possible under the assumption of shift-invariance (i.e., we assume uniform dyadic simplicial partitions of $\Omega = \mathbb{R}^d$, $\mathcal{V}_j = 2^{-j}\mathbb{Z}^d$, and that the HB-system is actually produced by translating and dilating $2^d - 1$ ψ-functions associated with the different edge directions). This assumption is typical for wavelet analysis, and allowed us in [LO96] to obtain a number of sharp results on the s-range for which the Riesz basis property holds for particular systems. Lack of space prevents us from presenting details on the theoretical tools used to produce these s-intervals. Instead we provide examples for the linear finite element case which have been considered in more detail in [LO96], and are often modeled after HB-constructions for bounded domains taken from the literature. The results reported here can be seen as qualitative information on the interior part of the associated HB-construction for domains. The Sobolev exponents of some practical importance are $s = 1$ and $s = 0$ (second order elliptic problems (including Helmholtz terms), Fredholm integral equations of second kind), as well as $s = \pm 1/2$ (boundary integral equations of first kind, interface problems in domain decomposition methods).

Details are given for $d = 2$. In this case, the grid is three-directional (as in Figure 1 a) below). The ψ-functions associated with the different edge directions (horizontal, vertical, and diagonal) will be labeled by h, v, and d. Whenever possible, we give the corresponding HB-construction for bounded polyhedral domains, and then specialize to the shift-invariant case. Furthermore, we set $\psi_{0,P} = \phi_{0,P}$ for all $P \in \mathcal{V}_0$.

Example 1 *Standard HB* (Yserentant [Yse86]). This system is given by

$$\psi_{j,P} = \phi_{j,P} \ , P \in \mathcal{W}_j \ , \ j \geq 1 \, ,$$

and is the simplest of all HB-systems.

Theorem 2 *The normalized standard HB-system $\tilde{\Psi}$ is a Riesz basis in $H^s(\Omega)$ if and only if $d/2 < s < 3/2$ $(d \leq 2)$.*

The case $s = 1$, $d = 2$, is not included here, in coincidence with the known fact [Yse86] that the standard HB-method of Yserentant is only suboptimal: $\kappa((D_J^{\Psi})^{-1} A_J^{\Psi}) \asymp J^2$ there.

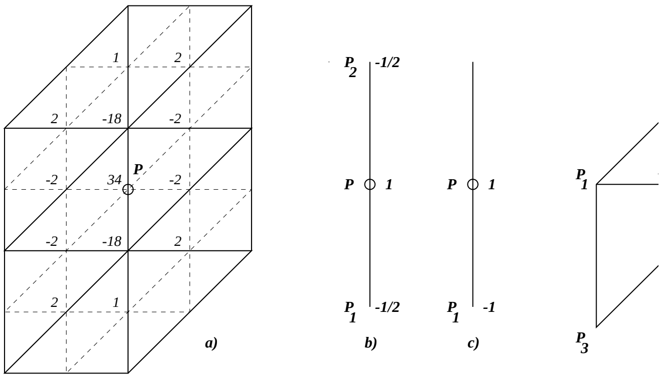
Example 2 *Extended NB system.* Though not fitting into the discussion of Riesz bases, we would like to mention the following interpretation of the optimality (see [Osw94], Section 4.2) of the BPX-preconditioner introduced by Bramble/Pasciak/Xu [BPX90]. We call $\Phi = \{\phi_{j,P} \, : \, P \in \mathcal{V}_j, \, j \geq 0\}$ an extended NB-system or BPX-system, and denote by $\tilde{\Phi}$ the corresponding $H^s(\Omega)$-normalized system. Note that Φ is not minimal. The finite sections Φ_J of this system obtained by taking only NB functions with $j \leq J$ are generating systems (not bases) for V_J. Nevertheless, we have

Theorem 3 *For arbitrary $d \geq 1$, the normalized BPX-system $\tilde{\Phi}$ is a frame in $H^s(\Omega)$ iff $0 < s < 3/2$.*

It should be mentioned that this simple enlargement of the standard HB-system not only improves the theoretical properties of the latter for $d \geq 2$. The practical performance (simplicity of implementation, operation count per preconditoning step, condition number bounds for H^1-problems) is surprisingly good.

Example 3 *L_2-semiorthogonal prewavelet systems* (Kotyczka/Oswald [Osw94, KO96], Junkherr [Jun94], Stevenson [Ste97b]). We call a HB-system an L_2-semiorthogonal prewavelet system if it is obtained by choosing the finite masks in (3) such that all $\psi_{j,P}$ are L_2-orthogonal to V_{j-1}, $j \geq 1$, and (5) still holds. It turns out (see [Osw94],

Figure 1 Masks for ψ_v: a) L_2-semiorthogonal prewavelet system, b) 3-point HB-system, c) 2-point HB-system, and d) notation for Example 5.



Section 4.4 for a similar argument) that the proof of Theorem 3 and the definition of Sobolev spaces with negative s by duality immediately imply

Theorem 4 *The normalized version $\tilde{\Psi}$ of an L_2 semiorthogonal prewavelet system is a Riesz basis for $H^s(\Omega)$ if $-3/2 < s < 3/2$.*

Thus, such systems cover most of the potential applications in an asymptotically optimal sense. For the shift-invariant case, an example with smallest possible support of the $\psi_{j,P}$ has been constructed in [KO96] for $d = 2$. Figure 1 a) shows the mask for ψ_v. The other two masks are obtained by suitable rotation. For general $\Omega \subset \mathbb{R}^2$, a mask construction has been proposed in [LO96], however, there is no rigorous proof of (5) for this case. Generalizations to $d \geq 3$ along the lines of [KO96, Jun94] do not seem to be of practical interest. One reason is the relatively large masks (e.g., an average of 29 non-zero coefficients is needed to satisfy the orthogonality constraint for $d = 3$ (uniform refinement case)). Very recently, Stevenson [Ste97b] came up with an alternative construction of L_2-semiorthogonal prewavelets which is suitable for all $d \leq 3$. However, the s-values of interest are also covered by much simpler HB-constructions (see below).

Finally let us mention that, from a theoretical (and practical) point of view, the construction of HB-systems consisting of functions $\psi_{j,P}$ which are orthogonal resp. semiorthogonal with respect to the variational scalar product $a(\cdot,\cdot)$ would be desirable. However, for $d \geq 2$ and the $H^s(\mathbb{R}^d)$ scalar product ($s > 0$) such systems cannot have uniformly bounded mask size, see [LO97b].

Example 4 *3-point HB-system* (Stevenson [Ste96, Ste97a]). If L_2-semiorthogonality is weakened, simplifications are possible. Let P_1, P_2 denote the endpoints of the edge in $\mathcal{T}_{j-1}$ which contains $P \in \mathcal{W}_j$ as midpoint. Set

$$\psi_{j,P} = \psi_{j,P} + a_1\psi_{j,P_1} + a_2\psi_{j,P_2}$$

and choose a_1, a_2 such that

$$\sum_{Q\in\mathcal{V}_j} \mu_{j,Q}\psi_{j,P}(Q)u_{j-1}(Q) = 0 \qquad \forall u_{j-1} \in V_{j-1} \ .$$

Here, the choice $\mu_{j,Q} = |\mathrm{supp}\,\phi_{j,Q}|/3$ guarantees that the corresponding quadrature formula $\sum_Q \mu_{j,Q}u(Q)$ is exact w. r. t. V_j. Thus, the construction can be interpreted as replacing L_2-orthogonality by discrete L_2-orthogonality.

This HB-system has been studied, both theoretically and numerically, by Stevenson [Ste96, Ste97a] for $d = 2, 3$. For the shift-invariant case (the masks for this case are edge- and d-independent, see Figure 1 b)), we showed in [LO96]

Theorem 5 *The normalized 3-point HB-system of Stevenson is a Riesz basis in $H^s(\mathbb{R}^d)$, $d \leq 3$, iff $-0.992036 < s < 3/2$.*

For partial results and numerical evidence in the case of general Ω, see [Ste96, Ste97a]. It turns out that in the shift-invariant case, one can construct a whole family of analogous, edge-oriented, HB-systems (see [LO96]Section 4.1, [LO97a]Section 3.2). The simplest one leads to a 2-point HB-system, see Figure 1 c) for the mask of ψ_v, with the corresponding s-interval still covering the L_2-case: $-0.044117 < s < 3/2$. It is not quite clear at the moment what the correct 2-point HB-definition is for general Ω.

Example 5 *Coarse-grid stabilized HB-systems* (Carnicer/Dahmen/P̃ena [CDP96], Vassilevski/Wang [VW97a, VW97b]). The common idea is to define

$$\psi_{j,P} = (Id_j - Q_{j-1})\phi_{j,P} \ , \quad P \in \mathcal{W}_j \ , \ j \geq 1 \ ,$$

where $Q_{j-1} : V_j \to V_{j-1}$, and Id_j denotes the identity on V_j. For Q_{j-1}, quasi-interpolant operators are suggested in [CDP96], Section 4.2, while [VW97b] prefers the use of approximations to the exact L_2-orthogonal projection obtained by approximately inverting the L_2-Gram matrix of the nodal basis in V_{j-1}. The most economical proposals from these papers lead to

$$\psi_{j,P} = \phi_{j,P} - \sum_{i=1}^{4} a_i\phi_{j-1,P_i} \ , \quad P \in \mathcal{W}_j \ , \tag{13}$$

where P_i denote the vertices of the two triangles in $\mathcal{T}_{j-1}$ sharing the edge e_P (with obvious modifications for P near the boundary). Compare Figure 1 d). As a rule all a_i are non-zero, thus, these proposals are essentially 5-point HB-systems.

For $d = 2$, the shift-invariant case was analyzed in [LO96], where we concentrated on the specific, one-parameter family of masks given by

$$a_1 = a_2 = a \ , \quad a_3 = a_4 = 1/8 - a \qquad (a \in \mathbb{R}) \ . \tag{14}$$

This class is remarkable in that the ψ-functions satisfy moment conditions of order 2, a property, which is desirable if stiffness matrix compression is an issue (e.g., for integral equation applications). The following table shows the s-range for which the scaled coarse-grid stabilized HB-system $\tilde{\Psi}$ specified by (13), (14), is a Riesz basis in $H^s(\mathbb{R}^d)$ for some a (the reader may view this as the last theorem of this note). In

Table 1 Coarse-grid stabilized HB-systems: Results

a-value	s-range	comments
5/48	$0.248994 < s < 3/2$	$m = \beta = 1$ in [VW97b]
1/8	$0.022818 < s < 3/2$	complexity as 3-point HB
1/6	$-0.357680 < s < 3/2$	[CDP96]
3/16	$-0.440765 < s < 3/2$	maximal s-range,[CS93]
1/4	$0.396793 < s < 3/2$	

[LO97a]Section 3.2, a more detailed table is given. E.g., we have found that for these systems the Riesz basis property holds in $L_2(\mathbb{R}^d)$ if $0.1271146 < a < 0.220647$ resp. in $H^1(\mathbb{R}^d)$ if $0.028759 < a < 0.3014364$. This shows a certain robustness of such constructions (provided that the moment conditions are preserved).

The intervals in Table 1 suggest that the counterparts for general Ω might well work, at least, for second order elliptic problems ($s = 1$). However, there are no definite results in this direction so far. Compare [VW97a] for theoretical results on the existence (with possibly quite large masks) of coarse-grid stabilized HB-Riesz bases for $H^s(\Omega)$, $s > 0$. A crude message from the above examples is that fine-grid corrected HB of simple structure seem to have better properties than coarse-grid stabilized HB-systems. However, the condition number computations presented below will slightly correct this impression.

4 Condition Numbers

The impression that a "larger s-interval" for the Riesz basis property to hold means "better practical performance" is misleading (though, by some kind of interpolation argument, one might expect good preconditioning effects if the s corresponding to a given variational problem is in the central part of the computed interval). On the other hand, when elliptic operators including parts of different order are the main concern, a large s-interval might be of benefit. In any case, numerical estimations of Riesz bounds resp. condition numbers $\kappa((D_J^\Psi)^{-1}A_J^\Psi)$ and other performance testing are recommended.

The following tables serve as an orientation for the more practically interested reader. We only present calculations of condition numbers $\kappa((D_J^\Psi)^{-1}A_J^\Psi)$ for $d = 2$ and $s = 0, 1$ (boundary integral equations, where $s = \pm 1/2$, are not addressed). The domain is the unit square, computations are done on standard uniform dyadic triangulations, and zero boundary conditions were imposed on the spaces V_j. The bilinear forms is the L_2-scalar product ($s = 0$) resp. is induced by the Laplace operator ($s = 1$). The index $j = 0$ corresponds to stepsize $h_0 = 1/2$, thus resulting in a one-dimensional V_0. The largest problems ($J = 7$) have dimension 65025. Everything else is implemented exactly as described above. For the 2-point HB-system, the choice of P_1 is to the right of and/or above P (compare Figure 1 c) for the notation). In the remaining boundary strip, the HB functions of Example 1 have been taken.

Table 2 H_0^1-case: Condition numbers $\kappa((D_J^\Psi)^{-1}A_J^\Psi)$

J	stHB	BPX	3ptHB	2ptHB	$a=5/48$	$a=1/8$	$a=1/6$	$a=3/16$
1	4.56	2.87	4.96	6.60	3.45	3.37	3.39	3.45
2	10.59	5.31	8.81	19.83	6.51	5.51	5.28	5.31
3	19.53	7.06	11.60	38.45	10.48	8.26	6.66	6.57
4	31.85	8.27	13.56	53.03	14.20	10.68	8.10	7.81
5	47.14	9.22	15.22	63.36	17.41	12.76	9.17	8.72
6	65.38	9.99	16.44	71.39	20.33	14.52	10.05	9.51
7	86.15	10.64	17.25	77.54	22.76	15.87	10.81	10.17

Table 3 L_2-case: Condition numbers $\kappa((D_J^\Psi)^{-1}A_J^\Psi)$

J	stHB	BPX	3ptHB	2ptHB	$a=5/48$	$a=1/8$	$a=1/6$	$a=3/16$
1	22	3.94	2.86	7.2	6.3	5.63	4.73	4.50
2	111	8.24	4.02	22.1	16.2	11.74	7.16	7.34
3	543	12.60	4.70	43.7	33.9	20.56	10.08	9.44
4	2565	16.75	5.10	72.5	60.3	30.73	12.26	11.37
5	11852	20.75	5.41	138.2	100.5	42.86	14.30	13.00
6	–	24.68	5.66	188.2	160.1	57.07	16.13	14.39

We finish with two observations. First, for all examples included in Tables 2 and 3 (standard HB, BPX, 3-point HB, 2-point HB, and coarse-grid stabilized HB for $a = 5/48$ (from [VW97b]), $a = 1/8$, $a = 1/6$ (from [CDP96]), and $a = 3/16$ (see [CS93])), the arithmetical costs per pcg-iteration would be almost the same if one neglects the overhead for computing masks. The pcg-step for the 5-point examples (Example 5) is asymptotically more expensive than the pcg-step only by a factor 1.2 for the cheapest method (standard HB). Similar considerations can be found in [Ste97b].

Secondly, when experimenting we found (for the first time) some HB-proposals which give for the H_0^1-case the same condition number behavior as in the BPX-method. This was interesting to us because so far all numerical evidence (also with other, wavelet based additive preconditioners) showed the superiority of the simple, frame-based BPX-algorithm by a factor of about 2, at least. Compare also the experiments in [VW97b] with different variants of coarse-grid stabilized HB-methods for $d = 2$ and $d = 3$. On the other hand, some cheap HB-proposals such as the 2-point HB-system which was mentioned in connection with Example 4, did not fulfill our expectations.

Further theoretical work and numerical testing is planned, in particular, for more general partitions, $d = 3$, and including the multiplicative (i.e., multigrid V-cycle) versions of the considered HB-methods. Another aspect which we wish to take up in the future is the design of HB-systems with small mask size and sufficiently many moment conditions for applications to integral equations where one wishes to cover the Sobolev spaces with $s = 0$ and $s = \pm1/2$. In the shift-invariant case, some simple proposals based on P0-elements have been discussed in [LO97a]. The methods of [LO96, LO97a] also allow similar investigations for C^1-elements.

REFERENCES

[BPX90] Bramble J. H., Pasciak J., and Xu J. (1990) Parallel multilevel preconditioners. *Math. Comp.* 55: 1–22.

[CDP96] Carnicer J. M., Dahmen W., and Peña J. M. (1996) Local decomposition of refinable spaces and wavelets. *Appl. Comput. Harm. Anal.* 3: 127–153.

[CS93] Cohen A. and Schlenker J.-M. (1993) Compactly supported biorthogonal wavelet bases with hexagonal symmetry. *Constr. Approx.* 9: 209–236.

[Dah96] Dahmen W. (1996) Multiscale analysis, approximation, and interpolation spaces. In Chui C. K. and Schumaker L. L. (eds) *Approximation Theory VIII, vol. 2*, pages 47–88. World Scientific, Singapore.

[Dau92] Daubechies I. (1992) *Ten Lectures on Wavelets*, volume 61 of *CBMS-NSF Reg. Conf. Ser. in Appl. Math.* SIAM Publication, Philadelphia.

[Jun94] Junkherr J. (1994) *Efficient Solution of Systems of Equations Resulting from Discretizing Weakly Singular Integral Equations of the First Kind (in German).* PhD thesis, Kiel University.

[KO96] Kotyczka U. and Oswald P. (1996) Piecewise linear prewavelets of small support. In Chui C. K. and Schumaker L. L. (eds) *Approximation Theory VIII, vol. 2*, pages 235–242. World Scientific, Singapore.

[LO96] Lorentz R. and Oswald P. (May 1996) Constructing economical Riesz bases for Sobolev spaces. Arbeitspapiere 993, GMD.

[LO97a] Lorentz R. and Oswald P. (March 1997) Criteria for hierarchical Riesz bases in Sobolev spaces. Arbeitspapiere 1059, GMD.

[LO97b] Lorentz R. and Oswald P. (1997) Nonexistence of compactly supported box spline prewavelets in Sobolev spaces. In LeMehaute A., Rabut C., and Schumaker L. L. (eds) *Surface Fitting and Multiresolution Methods*, pages 235–244. Vanderbuilt Univ. Press, Nashville.

[Osw94] Oswald P. (1994) *Multilevel Finite Element Approximation: Theory and Applications.* Teubner Skripten zur Numerik. Teubner, Stuttgart.

[Osw97] Oswald P. (1997) Multilevel solvers for elliptic problems on domains. In Dahmen W., Kurdila A. J., and Oswald P. (eds) *Multiscale Wavelet Methods for PDEs*, Wavelet Analysis and Its Applications. Academic Press, New York.

[Ste96] Stevenson R. (1996) A robust hierarchical preconditioner on general meshes. *Numer. Math.* to appear.

[Ste97a] Stevenson R. (1997) Experiments in 3d with a three-point hierarchical basis preconditioner. *Appl. Numer. Math.* to appear.

[Ste97b] Stevenson R. (January 1997) Piecewise linear (pre-)wavelets on non-uniform meshes. Report 9701, University of Nijmegen.

[VW97a] Vassilevski P. S. and Wang J. (1997) Stabilizing the hierarchical basis by approximate wavelets, I: Theory. *Numer. Linear Algebra Appl.* 4. to appear.

[VW97b] Vassilevski P. S. and Wang J. (1997) Stabilizing the hierarchical basis by approximate wavelets, II: Implementation and numerical results. *SIAM J. Sci. Comput.* submitted.

[Yse86] Yserentant H. (1986) On the multi-level splitting of finite element spaces. *Numer. Math.* 49: 379–412.

22

Overlapping Domain Decomposition for a Mixed Finite Element Method in Three Dimensions

Z. Cai, R.R. Parashkevov, T.F. Russell and X. Ye

1 Introduction

The work presented in this talk was motivated by the following question: Is it possible to find a computational technique for solving the linear system resulting from Mixed Finite Element discretizations of certain self-adjoint second order elliptic boundary value problems at the computational cost of a standard Galerkin FEM? There are two main obstacles to achieving the above stated goal. First, Mixed FEM formulations lead to a significantly larger number of unknowns compared to a standard conforming FEM of a comparable accuracy on the same triangulation of the domain. And secondly, the Mixed FEM produces a symmetric indefinite matrix problem as opposed to a symmetric and positive definite matrix in the standard FEM. The combined effect of these difficulties can often discourage end users from using Mixed Methods even in applications where a mixed approach can be beneficial (see [MSAC94]).

A number of researchers have studied this problem over the years and have contributed to developing several efficient iterative solvers. We acknowledge all their work, but for brevity, we will only consider in this talk approaches that involve Domain Decomposition ideas.

In [EW92], Ewing and Wang considered and analyzed a domain decomposition method for solving the discrete system of equations which result from mixed finite element approximation of second-order elliptic boundary value problems in two dimensions. The approach in [EW92] is first to seek a discrete velocity satisfying the discrete continuity equation through a variation of domain decomposition (static condensation), and then to apply a domain decomposition method to the reduced elliptic problem arising from elimination of the pressure and part of the velocity unknowns in the saddle-point problem. The crucial part of the approach in [EW92] is to characterize the divergence-free velocity subspaces. This is also the essential

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

difference with those in [GW87], [MR94], and [CMW95]. The Lagrange multipliers approach used in [GW87, CMW95] does produce a symmetric and positive definite matrix, but fails to address the other issue: the large number of unknowns. Recently, Chen, Ewing and Lazarov suggested in [CEL96] to reduce the number of unknowns by eliminating on a element by element basis the pressure variables. Then they applied a DD algorithm on the Lagrange multiplier variables only.

When comparing the two basic ideas (i.e. Lagrange multipliers and div-free subspace) one notes that they both re-formulate the original saddle-point problem into a symmetric and positive definite one and then apply some known DD algorithm. The difference is the number of unknowns in the discrete system. The dimension of the div-free velocity subspace is always smaller than the number of Lagrange multipliers.

In this paper, we will use the domain decomposition approach in [EW92] for the solution of the algebraic system resulting from the mixed finite element method applied to second-order elliptic boundary value problems in three dimensions. As mentioned above, the basis of the divergence-free velocity subspace plays an essential role in the approach; hence we will construct a basis of this subspace for the lowest-order rectangular Raviart-Thomas-Nedelec [RT77, Ned80] velocity space. The construction in two dimensions is rather easier than in three dimensions due to the fact that any divergence-free vector in 2-D can be expressed as the curl of a scalar stream function. Extension of this work to triangular or irregular meshes and to multilevel domain decomposition will be discussed in a forthcoming paper.

This approach has several practical advantages. For an $n \times n \times n$ grid in 3-D, the number of discrete unknowns is approximately $4n^3$, essentially one pressure and three velocity components per cell. Using the divergence-free subspace, we decouple the system in such a manner that the velocity can be obtained directly by solving a symmetric positive definite system of order roughly $2n^3$ thus coming closer than any other approach so far to the number of degrees of freedom in a standard Galerkin FEM. In contrast to some other proposed procedures, this does not require the introduction of Lagrange multipliers corresponding to pressures at cell interfaces, and it permits direct computation of the velocity, which is often the principal variable of interest, alone. If the pressure is also needed, it can be calculated inexpensively in an additional step. Furthermore, the approach deals readily with the case of full-tensor conductivity (cross-derivatives), where the mass matrix is fuller than tridiagonal and methods based on reduced integration (mass lumping) are difficult to apply. This case results, for example, from anisotropic permeabilities in flows in porous media, where highly discontinuous conductivity coefficients are also common. For such problems, mixed methods are known to produce more realistic velocities than standard techniques [MSAC94].

2 Mixed Finite Element Method

In this section, we begin with a brief review of the mixed finite element method with lowest-order Raviart-Thomas-Nedelec [RT77, Ned80] approximation space for second-order elliptic boundary value problems in three dimensions. For simplicity, we consider a homogeneous Neumann problem: find p such that

$$\left\{ \begin{array}{rcll} -\nabla \cdot (k\nabla p) & = & f, & \text{in} \quad \Omega = (0,\,1)^3, \\ (k\nabla p)\cdot \mathbf{n} & = & 0, & \text{on} \quad \partial\Omega, \end{array} \right. \tag{1}$$

where $f \in L^2(\Omega)$ satisfies the relation $\int_\Omega f = 0$, and $\mathbf{n}$ denotes the unit outward normal vector to $\partial\Omega$. The symbols $\nabla\cdot$ and ∇ stand for the divergence and gradient operators, respectively. Assume that $k = (k_{ij})_{3\times 3}$ is a given real-valued symmetric matrix function with bounded and measurable entries k_{ij} $(i,\, j = 1,\, 2,\, 3)$ and satisfies an ellipticity condition a.e. in Ω.

We shall use the following space to define the mixed variational problem. Let

$$H(div;\Omega) \equiv \{\mathbf{w} \in L^2(\Omega)^3 \,|\, \nabla\cdot\mathbf{w} \in L^2(\Omega)\},$$

which is a Hilbert space when equipped with the standard norm and the associated inner product. By introducing the flux variable

$$\mathbf{v} = -k\nabla p,$$

which is often of practical interest for many physical problems, we can rewrite the PDE of (1) as a first-order system

$$\left\{ \begin{array}{rcl} k^{-1}\mathbf{v} + \nabla p & = & 0, \\ \nabla\cdot\mathbf{v} & = & f, \end{array} \right.$$

and obtain the mixed formulation of (1): find $(\mathbf{v},\, p) \in \mathbf{V}\times\Lambda$ such that

$$\left\{ \begin{array}{lcll} a(\mathbf{v},\,\mathbf{w}) - b(\mathbf{w},\,p) & = & 0, & \forall\,\mathbf{w}\in\mathbf{V}, \\ b(\mathbf{v},\,\lambda) & = & (f,\,\lambda), & \forall\,\lambda\in\Lambda. \end{array} \right. \tag{2}$$

Here $\mathbf{V} = H_0(div;\Omega) \equiv \{\mathbf{w} \in H(div;\Omega)\,|\,\mathbf{w}\cdot\mathbf{n} = 0 \text{ on} \partial\Omega\}$, Λ is the quotient space $L_0^2(\Omega) = L^2(\Omega)/\{\text{constants}\}$, the bilinear forms $a(\cdot\,,\cdot) : \mathbf{V}\times\mathbf{V} \to I\!\!R$ and $b(\cdot\,,\cdot) : \mathbf{V}\times\Lambda \to I\!\!R$ are defined by

$$a(\mathbf{w},\,\mathbf{u}) = \int_\Omega (k^{-1}\mathbf{w})\cdot\mathbf{u}\,dx\,dy\,dz \quad \text{and} \quad b(\mathbf{w},\,\lambda) = \int_\Omega (\nabla\cdot\mathbf{w})\lambda\,dx\,dy\,dz$$

for any $\mathbf{w}$, $\mathbf{u}\in\mathbf{V}$ and $\lambda\in\Lambda$, respectively, and $(\cdot,\cdot)$ denotes the $L^2(\Omega)$ inner product.

To discretize the mixed formulation (2), we assume that we are given two finite element subspaces

$$\mathbf{V}^h \subset \mathbf{V} \quad \text{and} \quad \Lambda^h \subset \Lambda$$

defined on a uniform rectangular mesh with elements of size $O(h)$. The mixed approximation of $(\mathbf{v}\,, p)$ is defined to be the pair, $(\mathbf{v}^h,\, p^h) \in \mathbf{V}^h \times \Lambda^h$, satisfying

$$\left\{ \begin{array}{lcll} a(\mathbf{v}^h,\,\mathbf{w}) - b(\mathbf{w}\,,p^h) & = & 0, & \forall\,\mathbf{w}\in\mathbf{V}^h, \\ b(\mathbf{v}^h,\,\lambda) & = & (f,\,\lambda), & \forall\,\lambda\in\Lambda^h. \end{array} \right. \tag{3}$$

We refer to [RT77] for the definition of a class of approximation subspaces $\mathbf{V}^h$ and Λ^h. In this paper, we shall only consider the lowest-order R-T-N space defined on a rectangular triangulation of Ω. However, as shown in [CPRY95], this construction can readily be generalized to higher-order elements on non-orthogonal meshes and more general boundary conditions.

3 Domain Decomposition

Problem (3) is clearly symmetric and indefinite. To reduce it to a symmetric positive definite problem, we need a discrete velocity $\mathbf{v}_I^h \in \mathbf{V}^h$ satisfying

$$b(\mathbf{v}_I^h, \lambda) = (f, \lambda), \quad \forall\, \lambda \in \Lambda^h. \tag{4}$$

Define the discretely (as opposed to pointwise) divergence-free subspace $\mathbf{D}^h$ of $\mathbf{V}^h$:

$$\mathbf{D}^h = \{\mathbf{w} \in \mathbf{V}^h \,|\, b(\mathbf{w}, \lambda) = 0, \quad \forall\, \lambda \in \Lambda^h\}, \tag{5}$$

and let

$$\mathbf{v}_D^h = \mathbf{v}^h - \mathbf{v}_I^h,$$

which is obviously in $\mathbf{D}^h$ by the second equation of (3) and satisfies

$$a(\mathbf{v}_D^h, \mathbf{w}) = -a(\mathbf{v}_I^h, \mathbf{w}), \quad \forall\, \mathbf{w} \in \mathbf{D}^h, \tag{6}$$

by the first equation. This problem is symmetric and positive definite.

This suggests the following procedure to obtain $\mathbf{v}^h$, the solution of (3): find $\mathbf{v}_I^h \in \mathbf{V}^h$ satisfying (4), compute the projection $\mathbf{v}_D^h \in \mathbf{D}^h$ satisfying (6), then set $\mathbf{v}^h = \mathbf{v}_I^h + \mathbf{v}_D^h$. This procedure will be the basis for Algorithms 3.1 and 3.2 below. Given $\mathbf{v}_I^h$, (6) leads to a unique $\mathbf{v}^h$, which is independent of the choice of $\mathbf{v}_I^h$. For an $n \times n \times n$ grid, computing the projection $\mathbf{v}_D^h$ involves solving a SPD system of order approximately $2n^3$. Solving for p^h is optional; if it is desired, it can be obtained from the first equation in (3) once $\mathbf{v}^h$ is known.

There are many discrete velocities in $\mathbf{V}^h$ satisfying (4), and several approaches have been discussed in the literature for seeking such a discrete velocity (e.g., [EW92], [GW87], and [MR94]). All of these approaches are based on a type of domain decomposition (static condensation) method applied to problem (3). In a recent paper [CPRY95], we suggested a different approach which only requires solving a number of independent one-dimensional problems.

We shall use additive and multiplicative domain decomposition methods for approximate computation of the solution of problem (6). As usual, we first decompose the original domain Ω into non-overlapping subdomains $\Omega = \cup\tilde{\Omega}_j$, where each subdomain $\tilde{\Omega}_j$ has a diameter of size H and then extend generously (i.e. with overlap of order H) each $\tilde{\Omega}_j$. The restriction of any FE space to the coarse grid defined by the non-overlapping subdomains will be denoted by the index H, and the restriction to Ω_j by the index j. Next, we define the family of discretely divergence-free velocity subspaces $\{\mathbf{D}_j\}_{j=0}^J$ by $\mathbf{D}_0 = \mathbf{D}^H$, and for $j \in \{1, 2, ..., J\}$,

$$\mathbf{D}_j = \{\mathbf{u} \in \mathbf{V}_j \,|\, b(\mathbf{u}, \lambda) = 0, \quad \forall\, \lambda \in \Lambda_j\}.$$

For any $\mathbf{u} \in \mathbf{D}^h$, we define the projection operators $\mathbf{P}_j : \mathbf{D}^h \longrightarrow \mathbf{D}_j$ associated with the bilinear form $a(\cdot\,,\cdot)$ by

$$a(\mathbf{P}_j\mathbf{u}, \mathbf{w}) = a(\mathbf{u}, \mathbf{w}), \quad \forall\, \mathbf{w} \in \mathbf{D}_j,$$

for $j \in \{0, 1, ..., J\}$.

Algorithm 3.1 (Additive Domain Decomposition) *1. Compute* $\mathbf{v}_I^h \in \mathbf{V}^h$ *as in [CPRY95].*

2. Compute an approximation, $\mathbf{v}_D$*, of* $\mathbf{v}_D^h \in \mathbf{D}^h$ *by applying conjugate gradient iteration to*

$$\mathbf{P}\mathbf{v}_D = \mathbf{F} \tag{7}$$

where $\mathbf{P} = \mathbf{P}_0 + \mathbf{P}_1 + \cdots + \mathbf{P}_J$*,* $\mathbf{F} = \mathbf{F}_0 + \mathbf{F}_1 + \cdots + \mathbf{F}_J$*, and* $\mathbf{F}_j = \mathbf{P}_j \mathbf{v}_D^h$.

3. Set

$$\mathbf{v}^h = \mathbf{v}_D + \mathbf{v}_I^h.$$

Remark 3.1 *The right-hand side* $\mathbf{F}$ *in* (7) *can be computed by solving the coarse-grid problem and local subproblems. Specifically, for each* $j \in \{0, 1, ..., J\}$*,* $\mathbf{F}_j$ *is the solution of the following problem:*

$$a(\mathbf{F}_j, \mathbf{w}) = a(\mathbf{P}_j \mathbf{v}_D^h, \mathbf{w}) = -a(\mathbf{v}_I^h, \mathbf{w}), \quad \forall\, \mathbf{w} \in \mathbf{D}_j. \tag{8}$$

Algorithm 3.2 (Multiplicative Domain Decomposition) *1. Compute* $\mathbf{v}_I^h$ *as in the first step of* Algorithm 4.

2. Given an approximation $\mathbf{v}_D^l \in \mathbf{D}^h$ *to the solution* $\mathbf{v}_D^h$ *of* (6)*, define the next approximation* $\mathbf{v}_D^{l+1} \in \mathbf{D}^h$ *as follows:*

a) Set $W_{-1} = \mathbf{v}_D^l$.

b) For $j = 0, 1, ..., J$ *in turn, define* W_j *by*

$$W_j = W_{j-1} + \omega \mathbf{P}_j(\mathbf{v}_D^h - W_{j-1})$$

where the parameter $\omega \in (0, 2)$.

c) Set $\mathbf{v}_D^{l+1} = W_J$.

3. Set

$$\mathbf{v}^h = \mathbf{v}_I^h + \mathbf{v}_D^L.$$

Remark 3.2 $\mathbf{P}_j(\mathbf{v}_D^h - W_{j-1})$ *can be computed by solving the following problem:*

$$a(\mathbf{P}_j(\mathbf{v}_D^h - W_{j-1}), \mathbf{w}) = -a(\mathbf{v}_I^h + W_{j-1}, \mathbf{w}), \quad \forall\, \mathbf{w} \in \mathbf{D}_j. \tag{9}$$

A simple computation implies that the error propagation operator of multiplicative domain decomposition at the second step of Algorithm 3.2 has the form of

$$\mathbf{E} = (\mathbf{I} - \mathbf{P}_J)(\mathbf{I} - \mathbf{P}_{J-1}) \cdots (\mathbf{I} - \mathbf{P}_0). \tag{10}$$

Define a norm associated with the bilinear form $a(\cdot, \cdot)$ by

$$\|\mathbf{u}\|_a = a(\mathbf{u}, \mathbf{u})^{1/2}, \quad \forall\, \mathbf{u} \in \mathbf{D}^h.$$

We shall show in the last section that $\|\mathbf{E}\|_a$ is bounded by a constant which is less than one and independent of the mesh size h and the number of subdomains.

4 Construction of Divergence-Free Basis

Since the technique of the mixed method leads to a saddle-point problem which causes the final system to be indefinite, many well-established efficient linear system solvers cannot be applied. As we mentioned earlier, (3) could be symmetric and positive definite if we discretize it in the discrete divergence-free subspace $\mathbf{D}^h$. The construction of a basis for $\mathbf{D}^h$ is essential.

In this section, we will construct a computationally convenient basis for $\mathbf{D}^h$—the divergence-free subspace of $\mathbf{V}^h$. We will do this by first constructing a vector potential space $\mathbf{U}^h$ such that

$$\mathbf{D}^h = \mathbf{curl}\, \mathbf{U}^h. \tag{11}$$

Next, we will find a basis for $\mathbf{U}^h$ and we will define a basis for $\mathbf{D}^h$ by simply taking the curls of the vector potential basis functions.

Denote the mesh on $\Omega = (0,1)^3$ by $0 = x_0 < \cdots < x_i < \cdots < x_n = 1$, and similarly with y_j and z_k, $0 \leq j, k \leq n$. The assumption of the same number n of intervals in each direction is merely for convenience and is not necessary for the construction to follow. Let $\mathbf{U}^h$ be defined as follows:

$$\mathbf{U}^h = \operatorname{span} \left\{ \begin{pmatrix} \phi_i(y,z) \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \phi_j(x,z) \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ \phi_k(x,y) \end{pmatrix} \right\}, \tag{12}$$

where $1 \leq i \leq (n-1)^2$ (thus, only the first yz-slice is included) and $1 \leq j, k \leq n(n-1)^2$ (all xz- and xy-slices are included) and ϕ_i is the standard bi-linear nodal basis function associated with the edge i and is piece-wise constant in the third dimension. Note that the number of excluded ϕ_i's is $(n-1)^3$. If the number of intervals in the x-, y-, and z-directions were ℓ, m, and n, respectively, the number excluded would be $(\ell - 1)(m-1)(n-1)$, and would be the same if all but one xz- or xy-slice were excluded instead of all but one yz-slice.

Next, we list some properties of $\mathbf{U}^h$ which follow directly from the definition of the potential space.

Remark 4.1 $\mathbf{U}^h \not\subset H(div;\Omega)$ *, and hence,* $\mathbf{U}^h \not\subset H^1(\Omega)^3$.

Remark 4.2 *Every* $\Phi \in \mathbf{U}^h$ *satisfies* $\Phi \times \mathbf{n} = \mathbf{0}$ *on* $\partial\Omega$.

Remark 4.3 $\mathbf{U}^h$ *is locally divergence-free, i.e.* $\nabla \cdot \Phi = 0$ *on each element* $K \in \mathcal{T}^h$ *for every* $\Phi \in \mathbf{U}^h$.

Remark 4.4 $\mathbf{U}^h \subset H(curl;\Omega)$*, and hence* $\mathbf{curl}\, \mathbf{U}^h \subset \mathbf{V}^h$.

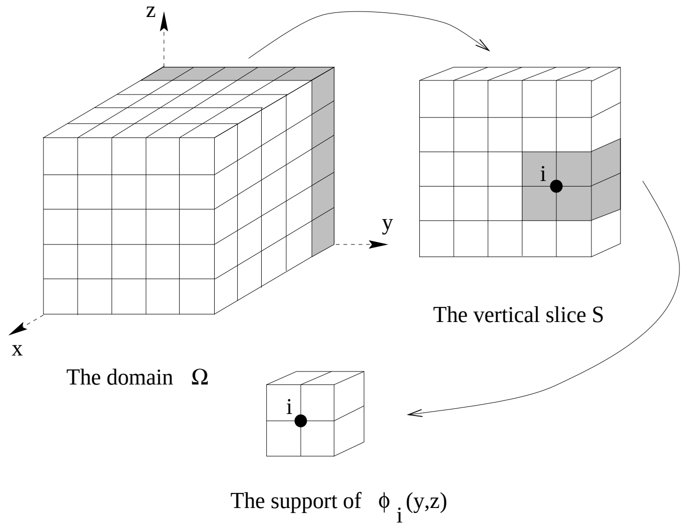
Since div $\mathbf{curl} \equiv 0$, we have $\mathbf{curl}\, \mathbf{U}^h \subset \mathbf{D}^h$. Counting dimensions,

$$\dim \mathbf{U}^h = (2n+1)(n-1)^2 = 2n^3 - 3n^2 + 1.$$

Also, div $\mathbf{V}^h$ consists of those piecewise constants with integral zero over Ω, hence has dimension $n^3 - 1$, and we obtain

$$\dim \mathbf{D}^h = \dim \mathbf{V}^h - \dim \operatorname{div} \mathbf{V}^h = 3(n-1)n^2 - (n^3 - 1) = 2n^3 - 3n^2 + 1.$$

Figure 1 The support of a typical potential basis function



We show in [CPRY95] that the curls of the vectors in (12) are linearly independent, so that

$$\dim \mathbf{D}^h = \dim \mathbf{curl}\, \mathbf{U}^h = \dim \mathbf{U}^h = 2n^3 - 3n^2 + 1,$$

which implies that for every divergence-free vector $\mathbf{v} \in \mathbf{D}^h$ there exists a unique potential vector $\Phi \in \mathbf{U}^h$ such that

$$\mathbf{v} = \mathbf{curl}\, \Phi.$$

The vector functions in (12) constitute only one possible choice of a basis for $\mathbf{U}^h$.

Remark 4.5 *The above-defined basis for $\mathbf{U}^h$ (and hence for $\mathbf{D}^h$) consists of vector functions with minimal possible support (4 elements).*

In [CPRY95] we prove the following Poincaré-type inequality:

Lemma 4.1 *There exists a constant $C(\Omega) > 0$ independent of the quasi-uniform mesh size h, such that for all $\Phi \in \mathbf{U}^h$ we have*

$$\|\Phi\|_{L^2(\Omega)^3} \leq C(\Omega)\, \|\mathbf{curl}\, \Phi\|_{L^2(\Omega)^3}. \tag{13}$$

(Since the vector potential space $\mathbf{U}^h \not\subset H^1(\Omega)^3$, inequality (13) does not follow from the standard Poincaré inequality.)

Corollary 4.2 *The linear system (6) to be solved in $\mathbf{D}^h$ has a symmetric and positive definite matrix with condition number of order $O(h^{-2})$.*

The result of the Lemma suggests that the curl semi-norm behaves like the H_0^1 semi-norm for scalar functions and thus allowing us to use fairly standard DD tools (as in [BPWX91, BX91, Cai93, DW87, Lio88, GR86]) to prove in [CPRY95] the following uniform convergence rate estimates:

Theorem 4.1 *For any vector* $\mathbf{v} \in \mathbf{D}^h$*, we have*

$$C_1 a(\mathbf{v}, \mathbf{v}) \leq a(\mathbf{P}\mathbf{v}, \mathbf{v}) \leq C_2 a(\mathbf{v}, \mathbf{v}) \tag{14}$$

where the positive constants C_1 *and* C_2 *are independent of* h *and* J.

Theorem 4.2 *The iterative method defined at the second step in* Algorithm 3.2 *is uniformly convergent, i.e.,*

$$\|\mathbf{E}\|_a \leq \gamma < 1 \tag{15}$$

where γ *is a constant that does not depend on the number of subdomains and the mesh size.*

Acknowledgement

The third author was supported in part by National Science Foundation Grant No. DMS-9312752, U.S. Army Research Office Grant No. 34593-GS, Ecodynamics Research Associates, Inc., Albuquerque, NM, and the U.S. Geological Survey, Water Resources Division, Denver, CO.

REFERENCES

[BPWX91] Bramble J. H., Pasciak J. E., Wang J., and Xu J. (1991) Convergence estimates for product iterative methods with applications to domain decomposition and multigrid. *Math. Comp.* 57: 1–21.

[BX91] Bramble J. H. and Xu J. (1991) Some estimates for a weighted L^2 projection. *Math. Comp.* 56: 463–476.

[Cai93] Cai Z. (1993) Norm estimates of product operators with application to domain decomposition. *Math. Comp.* 53: 251–276.

[CEL96] Chan Z., Ewing R. E., and Lazarov R. D. (1996) Domain decomposition algorithms for mixed methods for second-order elliptic problems. *Math. Comp.* 65: 467–490.

[CMW95] Cowsar L. C., Mandel J., and Wheeler M. F. (1995) Balancing domain decomposition for mixed finite elements. *Math. Comp.* 64: 989–1015.

[CPRY95] Cai Z., Parashkevov R. R., Russell T. F., and Ye X. (1995) Domain decomposition for a mixed finite element method in three dimensions. *SIAM J. Numer. Anal.* to appear.

[DW87] Dryja M. and Widlund O. (1987) An additive variant of the Schwarz alternating method for the case of many subregions. Tech. Rep. 339, Courant Institute.

[EW92] Ewing R. E. and Wang J. (1992) Analysis of the Schwarz algorithm for mixed finite element methods. *RAIRO Math. Modél. Anal. Numér.* 26: 739–756.

[GR86] Girault V. and Raviart P. A. (1986) *Finite Element Methods for Navier-Stokes Equations: Theory and Algorithms.* Springer-Verlag, New York.

[GW87] Glowinski R. and Wheeler M. F. (1987) Domain decomposition and mixed finite element methods for elliptic problems. In Glowinski R., Golub G. H., Meurant G. A., and Périaux J. (eds) *Proc. 1st Int. Symp. on Domain Decomposition Methods.* SIAM, Philadelphia.

[HY95] Hall C. and Ye X. (1995) Construction of null bases for the divergence operator associated with incompressible Navier-Stokes equations. *J. Linear Algebra and Applications* to appear.

[Lio87] Lions P. L. (1987) On the Schwarz alternating method I. In Glowinski R., Golub G. H., Meurant G. A., and Périaux J. (eds) *Proc. 1st Int. Symp. on Domain Decomposition Methods.* SIAM, Philadelphia.

[Mat89] Mathew T. F. (1989) *Domain decomposition and iterative refinement methods for mixed finite element discretizations of elliptic problems.* PhD dissertation, Courant Institute, New York.

[MSAC94] Mosé R., Siegel P., Ackerer P., and Chavent G. (1994) Application of the mixed hybrid finite element approximation in a groundwater flow model: Luxury or necessity? *Water Resour. Res.* 30: 3001–3012.

[Ned80] Nedelec J. C. (1980) Mixed finite elements in $I\!R^3$. *Numer. Math.* 35: 315–341.

[RT77] Raviart P. A. and Thomas J. M. (1977) *A mixed finite element method for 2nd order elliptic problems*, volume 606 of *Mathematical Aspects of Finite Element Methods, Lecture Notes in Mathematics*, pages 292–315. Springer-Verlag, New York.

23

Preconditioners for Mixed Spectral Element Methods for Elasticity and Stokes Problems

Luca F. Pavarino

1 Introduction: Linear Elasticity and Stokes Systems

We introduce and analyze some preconditioned iterative methods for the large indefinite linear systems arising from mixed spectral element discretizations of the linear elasticity and Stokes systems in three dimensions. For other approaches to the iterative solution of spectral element methods for Stokes and Navier-Stokes problems, see Maday, Patera and Rønquist [MPR92], Fischer and Rønquist [FR94], Rønquist [Røn96], Casarin [Cas96] and the references therein. For p-version finite element preconditioners for elasticity, see Mandel [Man96].

Let $\Omega \subset R^3$ be a polyhedral domain and Γ_0 a subset of its boundary. Let $\mathbf{V}$ be the Sobolev space $\mathbf{V} = \{\mathbf{v} \in H^1(\Omega)^3 : \mathbf{v}|_{\Gamma_0} = 0\}$. The linear elasticity problem consists in finding the displacement $\mathbf{u} \in \mathbf{V}$ of the domain Ω, fixed along Γ_0, subject to a surface force of density $\mathbf{g}$ along $\Gamma_1 = \partial\Omega - \Gamma_0$ and subject to an external force $\mathbf{f}$:

$$2\mu \int_\Omega \epsilon(\mathbf{u}) : \epsilon(\mathbf{v}) \; dx + \lambda \int_\Omega div\mathbf{u} \; div\mathbf{v} \; dx \;\; = < \mathbf{F}, \mathbf{v} > \quad \forall \mathbf{v} \in \mathbf{V}. \tag{1}$$

Here λ and μ are the Lamé constants, $\epsilon_{ij}(\mathbf{u}) = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i})$ is the linearized stress tensor, and the inner products are defined as $\epsilon(\mathbf{u}) : \epsilon(\mathbf{v}) = \sum_{i=1}^3 \sum_{j=1}^3 \epsilon_{ij}(\mathbf{u})\epsilon_{ij}(\mathbf{v})$, $< \mathbf{F}, \mathbf{v} >= \int_\Omega \sum_{i=1}^3 f_i v_i \; dx + \int_{\Gamma_1} \sum_{i=1}^3 g_i v_i \; ds$. When λ approaches infinity, this pure displacement model describes materials that are almost incompressible. In terms of the Poisson ratio $\nu = \frac{\lambda}{2(\lambda+\mu)}$, these materials are characterized by ν close to $1/2$. It is well known that when low order h-version finite elements are used in the discretization of (1), the locking phenomenon causes a deterioration of the convergence rate as $h \to 0$; see Babuška and Suri [BS92]. If the p-version is used instead, locking in $\mathbf{u}$ is eliminated, but it could still be present in quantities of interest such as $\lambda div\mathbf{u}$. Moreover, the stiffness matrix obtained by discretizing the pure displacement model (1) has a condition number that goes

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

to infinity when $\nu \to 1/2$. Therefore, the convergence rate of iterative methods deteriorates rapidly as the material becomes almost incompressible. Locking problems are eliminated altogether by introducing the new variable $p = -\lambda div \mathbf{u} \in L^2(\Omega) = W$ and by rewriting the pure displacement problem in a mixed formulation (see Brezzi and Fortin [BF96]): Find $(\mathbf{u}, p) \in \mathbf{V} \times W$ such that

$$\left\{ \begin{array}{lclcl} a(\mathbf{u},\mathbf{v}) & + & b(\mathbf{v},p) & = & <\mathbf{F},\mathbf{v}> \quad \forall \mathbf{v} \in \mathbf{V} \\ b(\mathbf{u},q) & - & \frac{1}{\lambda}c(p,q) & = & 0 \qquad \forall q \in W, \end{array} \right. \tag{2}$$

where $a(\mathbf{u},\mathbf{v}) = 2\mu \int_\Omega \epsilon(\mathbf{u}) : \epsilon(\mathbf{v}) dx$, $b(\mathbf{v},q) = -\int_\Omega div \mathbf{v} q dx$, $c(p,q) = \int_\Omega pq dx$. When $\lambda \to \infty$ (or, equivalently, $\nu \to 1/2$), we obtain from (2) the limiting problem for incompressible elasticity. In case of homogeneous Dirichlet boundary conditions on the whole boundary $\partial\Omega$, problem (2) is equivalent to a generalized Stokes problem, with $a(\cdot,\cdot)$ replaced by $\overline{a}(\mathbf{u},\mathbf{v}) = \mu \int_\Omega \nabla \mathbf{u} : \nabla \mathbf{v} dx$ and with $c(\cdot,\cdot)$ scaled by $\lambda + \mu$ instead of λ. In this case, the pressure will have zero mean value, so we define $W = L_0^2(\Omega)$. When $\lambda \to \infty$ we obtain the classical Stokes system describing the velocity $\mathbf{u}$ and pressure p of a fluid of viscosity μ: Find $(\mathbf{u}_0, p_0) \in \mathbf{V} \times W$ such that

$$\left\{ \begin{array}{lclcl} \overline{a}(\mathbf{u}_0,\mathbf{v}) & + & b(\mathbf{v},p_0) & = & <\mathbf{F},\mathbf{v}> \quad \forall \mathbf{v} \in \mathbf{V} \\ b(\mathbf{u}_0,q) & & & = & 0 \qquad \forall q \in W. \end{array} \right. \tag{3}$$

2 Mixed Spectral Element Methods

Let Ω_{ref} be the reference cube $[-1,1]^3$, $Q_n(\Omega_{ref})$ be the set of polynomials on Ω_{ref} of degree n in each variable and $P_n(\Omega_{ref})$ be the set of polynomials on Ω_{ref} of total degree n. Let the domain Ω be decomposed into a finite element triangulation $\bigcup_{i=1}^N \Omega_i$ of nonoverlapping elements. Each Ω_i is the affine image of the reference cube $\Omega_i = F_i(\Omega_{ref})$, where F_i is an affine mapping. We discretize each displacement component by conforming spectral elements, i.e. by continuous, piecewise polynomials of degree n:

$$\mathbf{V}^n = \{\mathbf{v} \in \mathbf{V} : v_k|_{\Omega_i} \circ F_i \in Q_n(\Omega_{ref}),\ i = 1, \cdots, N, \quad k = 1,2,3\}.$$

We consider two choices for the discrete pressure space W^n:

$$\begin{array}{lcl} W_1^n & = & \{q \in W : q_i \circ F_i \in Q_{n-2}(\Omega_{ref}),\ i = 1, \cdots, N\}, \\ W_2^n & = & \{q \in W : q_i \circ F_i \in P_{n-1}(\Omega_{ref}),\ i = 1, \cdots, N\}. \end{array}$$

The first choice gives us the $Q_n - Q_{n-2}$ method that Maday, Patera and Rønquist [MPR92] proposed for the Stokes system. A very convenient basis for W_1^n consists of the tensor-product Lagrangian interpolants associated with the internal Gauss-Lobatto-Legendre (GLL) nodes, described in the next section. The second choice gives us the Method 2 analyzed in Stenberg and Suri [SS96]. For this space standard p-version bases can be used. We will call this method $Q_n - P_{n-1}$.

Gauss-Lobatto-Legendre (GLL) Quadrature and the Discrete Problem

Denote by $\{\xi_i, \xi_j, \xi_k\}_{i,j,k=0}^n$ the set of GLL points on $Q_n(\Omega_{ref})$, and by σ_i the weight associated with ξ_i . Let $l_i(x)$ be the Lagrange interpolating polynomial vanishing at all the GLL nodes except at ξ_i, where it equals one. The basis functions on the reference cube are then defined by a tensor product as $l_i(x)l_j(y)l_k(z)$, $0 \leq i,j,k \leq n$. This is a nodal basis, since every polynomial in $Q_n(\Omega_{ref})$ can be written as $u(x,y,z) = \sum_{i=0}^{n}\sum_{j=0}^{n}\sum_{k=0}^{n} u(\xi_i,\xi_j,\xi_k)l_i(x)l_j(y)l_k(z)$. We then replace each integral of the continuous model (2) by GLL quadrature sums:

$$(u,v)_{Q,\Omega} = \sum_{s=1}^{N}\sum_{i,j,k=0}^{n} (u \circ F_s)(v \circ F_s)|J_s|(\xi_i,\xi_j,\xi_k)\sigma_i\sigma_j\sigma_k,$$

where $|J_s|$ is the determinant of the Jacobian of F_s. The analysis of this discretization technique can be found in Bernardi and Maday [BM92] and Maday, Patera and Rønquist [MPR92]. Applying this spectral element discretization to (2), we obtain the following discrete elasticity problem: Find $(\mathbf{u},p) \in \mathbf{V}^n \times W^n$ such that

$$\left\{ \begin{array}{lclcll} a_Q(\mathbf{u},\mathbf{v}) & + & b_Q(\mathbf{v},p) & = & < \mathbf{F},\mathbf{v} >_{Q,\Omega} & \forall \mathbf{v} \in \mathbf{V}^n \\ b_Q(\mathbf{u},q) & - & \frac{1}{\lambda}c_Q(p,q) & = & 0 & \forall q \in W^n, \end{array} \right. \tag{4}$$

where $a_Q(\mathbf{u},\mathbf{v}) = 2\mu(\epsilon(\mathbf{u}) : \epsilon(\mathbf{v}))_{Q,\Omega}$, $b_Q(\mathbf{v},q) = -(div\mathbf{v},q)_{Q,\Omega}$, $c(p,q) = (p,q)_{Q,\Omega}$. This is a saddle point problem with a penalty term and has the following matrix form:

$$Kx = \begin{bmatrix} A & B^T \\ B & -\frac{1}{\lambda}C \end{bmatrix} x = b\ . \tag{5}$$

The stiffness matrix K is symmetric and indefinite. It is less sparse than the one obtained by low-order finite elements, but is still well-structured. In the incompressible case, the C block is zero. For the Stokes problem, the discretization of the equivalent formulations (3) leads to an analogous block structure, with A consisting of three uncoupled discrete Laplacians.

The inf-sup Constant for Spectral Elements

The convergence of mixed methods depends not only on the approximation properties of the discrete spaces $\mathbf{V}^n$ and W^n, but also on a stability condition known as the inf-sup (or LBB) condition; see Brezzi and Fortin [BF96]. While many important h-version finite elements for Stokes problems satisfy the inf-sup condition with a constant independent of h, the important spectral elements proposed for Stokes problems, such as the $Q_n - Q_{n-2}$ and $Q_n - P_{n-1}$ methods, have an inf-sup constant that approaches zero as $n^{-(d-1)/2}$ $(d = 2,3)$. This result has been proven for the $Q_n - Q_{n-2}$ method by Maday, Patera and Rønquist [MPR92], where an example is constructed showing that this estimate is sharp. Stenberg and Suri [SS96] proved the following, more general, result covering both methods.

Theorem 1. (Stenberg and Suri [SS96]) Let the spaces $\mathbf{V}^n$ *and* W^n *satisfy assumptions (A1)-(A4) of [SS96] (satisfied by both our methods). Then for* $d = 2,3$

$$\sup_{\mathbf{v}\in\mathbf{V}^n\setminus\{0\}} \frac{(div\mathbf{v},q)}{\|\mathbf{v}\|_{H^1}} \geq Cn^{-(\frac{d-1}{2})}\|q\|_{L^2} \quad \forall q \in W^n,$$

where the constant C *is independent of* n, N *and* q.

In matrix form, the inf-sup condition becomes $q^tBA^{-1}B^tq \geq \beta_0^2 q^tCq$, $\forall q \in W^n$,

Table 1 Substructuring preconditioner: local condition numbers of $\hat{S}^{-1}S$

n	ν						
	0.3	0.4	0.49	0.499	0.4999	0.49999	0.499999
8	59.6995	64.5997	122.126	176.323	187.449	188.659	188.781

ν	n						
	2	3	4	5	6	7	8
0.499999	-	60.3303	89.1704	112.048	137.546	162.999	188.781

where $\beta_0 = Cn^{-(\frac{d-1}{2})}$ is the inf-sup constant of the method. Therefore β_0^2 scales as $\lambda_{min}(C^{-1}BA^{-1}B^t)$. Numerical experiments by Maday, Patera and Rønquist [MPR92], have shown that for the $Q_n - Q_{n-2}$ method, for practical values of n (e.g. $n \leq 16$), the dependence of β_0 on n is much weaker. Our numerical experiments show that the situation is even better for the $Q_n - P_{n-1}$ method. The trade-off in this case is the loss of a tensorial basis.

3 Preconditioned Iterative Methods

We will consider three classes of preconditioners: a) block-diagonal and b) triangular preconditioners for the whole indefinite system $Kx = b$; and c) substructuring methods for the Schur complement S of K associated with the interface variables. a) and b) are based on recent work by Klawonn [Kla96] on standard h-version finite elements, while c) is based on the wire basket spectral element methods introduced by Pavarino and Widlund [PW96] for the scalar case.

Block-diagonal Preconditioners

Consider the block-diagonal preconditioner with positive definite blocks $\hat{A}$ and $\hat{C}$:

$$\hat{D} = \begin{bmatrix} \hat{A} & 0 \\ 0 & \hat{C} \end{bmatrix} . \tag{6}$$

$\hat{A}$ and $\hat{C}$ are assumed to be good preconditioners for A and C respectively:
i) $\exists a_0, a_1 > 0$ such that $a_0^2 \mathbf{v}^t \hat{A} \mathbf{v} \leq \mathbf{v}^t A \mathbf{v} \leq a_1^2 \mathbf{v}^t \hat{A} \mathbf{v}, \quad \forall \mathbf{v} \in \mathbf{V}^n$;
ii) $\exists c_0, c_1 > 0$ such that $c_0^2 q^t \hat{C} q \leq q^t C q \leq c_1^2 q^t \hat{C} q, \quad \forall q \in W^n$. Interesting choices for $\hat{A}$ are given by h-version finite element discretizations on the GLL mesh or by substructuring domain decomposition methods, where a_0 and a_1 have a polylogarithmic dependence on the spectral degree n (for the scalar case, see Pavarino and Widlund [PW96] and Casarin [Cas96]). Since the resulting preconditioned system is symmetric, we can use the Preconditioned Conjugate Residual Method (PCR); see Hackbusch [Hac94]. Combining Klawonn's result ([Kla96], pp. 46-47) and Theorem 1, we obtain the following convergence result.
Theorem 2. If K is the stiffness matrix of the discrete system (4) obtained with either the $Q_n - Q_{n-2}$ or the $Q_n - P_{n-1}$ method and $\hat{D}$ is the block-diagonal preconditioner

Table 2 Exact block-diagonal preconditioner: iteration counts for $Q_n - Q_{n-2}$ on one element (in brackets are the iterations counts with the inexact Q_1 **u**-block and exact p-block)

n	ν							
	0.3	0.4	0.49	0.499	0.4999	0.49999	0.499999	0.5
9	11 (57)	15 (72)	31 (139)	39 (173)	41 (179)	41 (179)	41 (179)	41 (179)

ν	n							
	2	3	4	5	6	7	8	9
0.5	1 (1)	7 (13)	21 (48)	31 (84)	35 (111)	37 (134)	39 (158)	41 (179)

(6), then

$$cond(\hat{D}^{-1}K) \leq C\beta_0^{-1} = Cn^{(\frac{d-1}{2})}, \qquad d = 2, 3.$$

This implies that the number of iterations of our algorithm is bounded by $Cn^{(\frac{d-1}{2})}$.

Triangular Preconditioners

Consider the lower and upper triangular preconditioners

$$\hat{T}_L = \begin{bmatrix} \hat{A} & 0 \\ B & \hat{C} \end{bmatrix}, \qquad \hat{T}_U = \begin{bmatrix} \hat{A} & B^T \\ 0 & \hat{C} \end{bmatrix}, \tag{7}$$

where $\hat{A}$ and $\hat{C}$ are positive definite matrices. We will denote by T_L and T_U the case with exact blocks $\hat{A} = A$ and $\hat{C} = C$. Since the resulting preconditioned system is no longer symmetric or positive definite, we need to use Krylov methods for general nonsymmetric systems. We will consider three relatively recent methods: GMRES, Bi-CGSTAB and QMR; see Freund, Golub and Nachtigal [FGN92]. We remark that each application of the inverse of the triangular preconditioners $\hat{T}_L$ or $\hat{T}_U$ is only marginally more expensive than the block-diagonal preconditioner, because in addition to the solution of a system for $\hat{A}$ and one for $\hat{C}$, it requires only one application of B (or B^t). Klawonn ([Kla96], p. 56) proved that the spectrum of $T^{-1}K$ is real and positive. Combining Klawonn's result and Theorem 1, we obtain the following result.

Theorem 3. If K is the stiffness matrix of the discrete system (4) obtained with either $Q_n - Q_{n-2}$ or $Q_n - P_{n-1}$ spectral elements and T is the lower or upper triangular preconditioner (7) with exact blocks , then

$$cond(T^{-1}K) \leq C\beta_0^{-2} = Cn^{(d-1)}, \qquad d = 2, 3.$$

The case of a triangular preconditioner with inexact blocks is studied in Theorem 5.2 in Klawonn [Kla96], pg. 59, under the standard assumptions i) and ii) of the previous section. The estimate provided is analog to the case with exact blocks, but it is more complicated and we refer to [Kla96] for the details.

A Substructuring Preconditioner

For scalar elliptic problems, a complete study of substructuring methods for h-version finite elements can be found in Dryja, Smith and Widlund [DSW94]. For the spectral

Table 3 Exact block-diagonal preconditioner: iteration counts for $Q_n - Q_{n-2}$ on many elements

n	$N = N_x \times N_y \times N_z$	ν							
		0.3	0.4	0.49	0.499	0.4999	0.49999	0.499999	0.5
2	$8 = 2 \times 2 \times 2$	6	7	7	7	7	7	7	7
2	$64 = 4 \times 4 \times 4$	10	13	19	21	21	21	21	21
2	$216 = 6 \times 6 \times 6$	10	13	21	23	23	23	23	23

Table 4 Exact lower-triangular preconditioner: iteration counts for $Q_n - Q_{n-2}$ on one element; G=GMRES, B=Bi-CGSTAB, Q=QMR

n	ν							
	0.3	0.4	0.49	0.499	0.4999	0.49999	0.499999	0.5
	G B Q	G B Q	G B Q	G B Q	G B Q	G B Q	G B Q	G B Q
11	7 4 6	9 5 8	21 14 18	28 23 24	29 32 25	29 25 25	29 23 25	29 23 25

ν	n							
	4	5	6	7	8	9	10	11
	G B Q	G B Q	G B Q	G B Q	G B Q	G B Q	G B Q	G B Q
0.5	11 10 12	19 14 18	21 15 20	22 20 21	24 20 22	26 17 23	28 20 25	29 23 25

element case, see Pavarino and Widlund [PW96] and Casarin [Cas96]. If we order first all interior variables and then all the interface variables, $(\mathbf{u}_I, p, \mathbf{u}_B)$ (we recall that pressure unknowns are only interior), the stiffness matrix K can be reordered in the block form

$$K = \begin{pmatrix} K_{II} & K_{IB} \\ K_{IB}^T & K_{BB} \end{pmatrix}.$$

Eliminating the interior variables, we are left with the solution of a linear system with the Schur complement $S = K_{BB} - K_{IB}^T K_{II}^{-1} K_{IB}$. Our substructuring method will define a preconditioner for S. We further subdivide the interface variables into face and wire basket variables $\mathbf{u}_B = (\mathbf{u}_{\mathcal{F}}, \mathbf{u}_{\mathcal{W}})$, so that S can be reordered in the block form

$$S = \begin{pmatrix} S_{\mathcal{F}\mathcal{F}} & S_{\mathcal{F}\mathcal{W}} \\ S_{\mathcal{F}\mathcal{W}}^T & S_{\mathcal{W}\mathcal{W}} \end{pmatrix}.$$

Our additive preconditioner $\hat{S}$ is built from independent solvers associated with each face $\mathcal{F}_i$ (local problems) and the wire basket $\mathcal{W}$ (coarse problem):

$$\hat{S}^{-1} = \sum_{faces \mathcal{F}_i} R_{\mathcal{F}_i}^T S_{\mathcal{F}_i\mathcal{F}_i}^{-1} R_{\mathcal{F}_i} + R_0^T S_{\mathcal{W}\mathcal{W}}^{-1} R_0,$$

where R_0 represent a change of basis for $\mathcal{W}$ and $R_{\mathcal{F}_i}$ are restrictions matrices. Each local solver $S_{\mathcal{F}_i\mathcal{F}_i}^{-1}$ and $S_{\mathcal{W}\mathcal{W}}^{-1}$ can be replaced by an appropriate approximate solver. In joint work with O. Widlund, we are in the process of analyzing this algorithm using the Schwarz framework and recent work by Casarin [Cas96] for Stokes problems.

Table 5 Exact lower-triangular preconditioner: iteration counts for $Q_n - Q_{n-2}$ on many elements; G=GMRES, B=Bi-CGSTAB, Q=QMR

n	N	ν							
		0.3	0.4	0.49	0.499	0.4999	0.49999	0.499999	0.5
		G B Q	G B Q	G B Q	G B Q	G B Q	G B Q	G B Q	G B Q
2	2^3	3 2 4	4 2 4	4 3 4	4 3 4	4 3 4	4 3 4	4 3 4	4 3 4
2	4^3	5 4 6	6 4 7	9 6 10	10 7 11	10 7 11	10 7 11	10 7 11	10 7 11
2	6^3	6 4 6	7 5 7	10 7 11	11 7 12	11 7 12	11 7 12	11 7 12	11 7 12

4 Numerical Results

All the computations were performed in MATLAB 4.2 on Sun SPARC stations. The model problem considered is (2) on the reference cube $[-1,1]^3$, discretized with the $Q_n - Q_{n-2}$ or $Q_n - P_{n-1}$ spectral element methods. The resulting discrete systems have a matrix structure as in (5). The iterative methods considered are PCR for the block-diagonal preconditioner and GMRES (without restart), Bi-CGSTAB and QMR for the triangular preconditioner. The initial guess is zero and the right-hand side consists of uniformly distributed random numbers in [-1,1]. The stopping criterion is $\|r_i\|_2/\|r_0\|_2 \leq 10^{-6}$, where r_i is the $i-$th residual. We considered mainly preconditioners with exact blocks, in order to study the algorithms under the best of circumstances (inexact $\mathbf{u}$-blocks based on piecewise linear Q_1 finite elements on the GLL mesh are considered in Table 2). For brevity, we report only the results for the $Q_n - Q_{n-2}$ method. The $Q_n - P_{n-1}$ iteration counts were consistently better, thanks to a better inf-sup constant. More details for the block-diagonal and triangular preconditioners can be found in Pavarino [Pav96a], [Pav96b].

The results reported in the following tables agree with the theory: the convergence rate of the proposed methods is independent of ν and N but is mildly dependent on n (almost linearly for incompressible materials and Stokes problems) via the inf-sup constant.

REFERENCES

[BF91] Brezzi F. and Fortin M. (1991) *Mixed and Hybrid Finite Element Methods.* Springer-Verlag, Berlin.

[BM92] Bernardi C. and Maday Y. (1992) *Approximations Spectrales de Problèmes aux Limites Elliptiques.* Springer-Verlag France, Paris.

[BS92] Babuška I. and Suri M. (1992) Locking effects in the finite element approximation of elasticity problems. *Numer. Math.* 62: 439–463.

[Cas96] Casarin M. (1996) *Schwarz Preconditioners for Spectral and Mortar Finite Element Methods with Applications to Incompressible Fluids.* PhD thesis, Dept. of Math., Courant Institute, New York University.

[DSW94] Dryja M., Smith B. F., and Widlund O. B. (1994) Schwarz analysis of iterative substructuring algorithms for elliptic problems in three dimensions. *SIAM J. Numer. Anal.* 31(6): 1662–1694.

[FGN92] Freund R., Golub G. H., and Nachtigal N. (1992) *Iterative Solution of Linear Systems*, pages 57–100. Acta Numerica. Cambridge University Press.

[FR94] Fischer P. and Rønquist E. (1994) Spectral element methods for large scale parallel Navier-Stokes calculations. *Comp. Meths. Appl. Mech. Eng.* 116: 69–76.

[Hac94] Hackbusch W. (1994) *Iterative Solution of Large Sparse Systems of Equations.* Springer-Verlag, Berlin.

[Kla96] Klawonn A. (1996) *Preconditioners for Indefinite Problems.* PhD thesis, Westfälische Wilhelms-Universität Münster, Angewandte Mathematik und Informatik. Tech. Rep. 8/96-N.

[Man96] Mandel J. (1996) Iterative methods for p-version finite elements: preconditioning thin solids. *Comp. Meths. Appl. Mech. Eng.* 133: 247–257.

[MPR92] Maday Y., Patera A., and Rønquist E. (1992) The $P_N \times P_{N-2}$ method for the approximation of the Stokes problem. Technical Report 92009, Dept. of Mech. Engr., M.I.T.

[Pav96a] Pavarino L. F. (1996) Preconditioned conjugate residual methods for mixed spectral discretizations of elasticity and Stokes problems. Technical Report I.A.N.-CNR 988, Istituto di Analisi Numerica del CNR, Pavia, Italy. To appear in Comp. Meths. Appl. Mech. Eng.

[Pav96b] Pavarino L. F. (1996) Preconditioned mixed spectral element methods for elasticity and Stokes problems. Technical Report I.A.N.-CNR 1006, Istituto di Analisi Numerica del CNR, Pavia, Italy. To appear in SIAM J. Sci. Comp.

[PW96] Pavarino L. F. and Widlund O. B. (1996) A polylogarithmic bound for an iterative substructuring method for spectral elements in three dimensions. *SIAM J. Numer. Anal.* 33(4): 1303–1335.

[Røn96] Rønquist E. (1996) A domain decomposition solver for the steady Navier-Stokes equations. In Ilin A. and Scott L. (eds) *Proc. of ICOSAHOM '95.*

[SS96] Suri M. and Stenberg R. (1996) Mixed *hp* finite element methods for problems in elasticity and Stokes flow. *Numer. Math.* 72(3): 367–390.

24

Algebraic Domain Decomposition Method for Unstructured Grids

Yair Shapira

1 Introduction

The Black Box Multigrid method of [Den82] is considered robust for diffusion problems with variable coefficients. However, it is applicable only to structured grids, that is, 3^d-coefficient stencils (where d is the dimension of the problem) and not to more complicated (e.g., unstructured) finite element schemes. Furthermore, it is pointed out in [Sha94] [Sha96a] that Black Box Multigrid stagnates for certain diffusion problems with high diffusion areas separated by thin strips. Surprisingly, this stagnation occurs when the discontinuity curves are aligned with all the coarse grids, case which can be handled easily by either standard multigrid or the method of [BPWX91]. The AutoMUG method of [Sha94] [Sha96a] avoids this stagnation but diverges for other examples. In [Sha96b] this stagnation is studied and a modified version of Black Box Multigrid which avoids it is introduced. This version is related to the method of [KM81] and is based on 'throwing' certain matrix elements onto the main diagonal when constructing the prolongation operator from coarse to fine grids. It is shown in [Sha96b] that this version is robust both theoretically (for a certain class of problems) and numerically (for the above example and others). The method is generalized in [Sha97] to finite element schemes on locally refined meshes.

In this work we further generalize the method to finite element schemes on unstructured meshes which do not necessarily arise from local refinement. The method is based on a given domain decomposition, usually determined by the physical or the geometrical nature of the problem (Figure 1). The idea is to choose suitable vertex variables on the interface between subdomains for serving as a coarse grid. The prolongation operator extending a coarse grid function to the whole grid consists of two steps: first solve low order systems for defining the function on the interfaces between subdomains; then solve the original scheme on each subdomain separately for defining the function in the subdomain interiors. (For simplicity we consider the 2-d case; in 3-d, three steps are needed.) The coarse grid equation is obtained from a

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Figure 1 The unstructured grid and the domain decomposition.

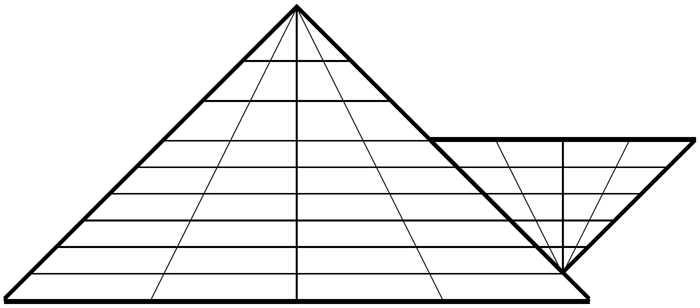
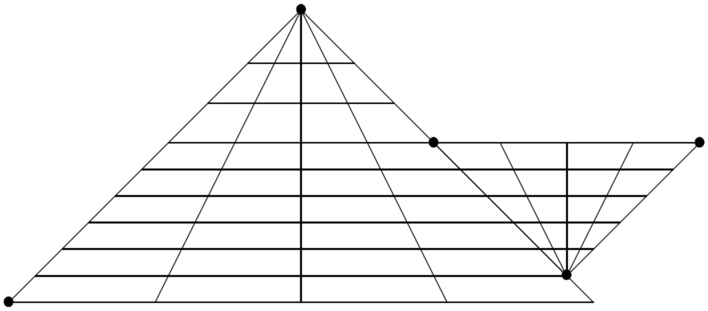


Figure 2 The coarse grid variables are denoted by '•'.



Galerkin scheme; it is solved either directly or iteratively using some preconditioning method or multigrid. Both the formulation of the coarse grid equation and the actual restriction and prolongation use only local operations which can be done in parallel. The method may be supplemented with presmoothing and postsmoothing as in multigrid or with an outer acceleration method. Unlike in [BPWX91], it is not assumed here that the discontinuities in the coefficients of the PDE are aligned with the coarse mesh. The method and the analysis are algebraic in the sense that, once the domain decomposition and the coarse grid are determined, only the coefficient matrix is used and not the PDE or the finite element mesh. Therefore, the method is named Algebraic Domain Decomposition (ADD).

2 The Algebraic Domain Decomposition Method

Consider a finite element scheme for an elliptic boundary value problem on a mesh of the type used in [BPWX91] (illustrated in Figure 1). Assume that the underlying linear system is given by

$$Ax = \mathbf{b}, \tag{1}$$

where x is the vector of unknowns corresponding to the nodes in the mesh, $\mathbf{b}$ is the right-hand side vector and A is the nonsingular coefficient matrix.

Consider a domain decomposition as in Figure 1, where nodes on the thick lines correspond to interface or boundary unknowns. Let some of these unknowns (typically, vertex variables such as those denoted by '•' in Figure 2) serve as coarse grid variables. In the following we denote by c the set of coarse grid variables, by b the set of the other boundary and interface variables and by s the set of all other variables (corresponding to nodes in subdomain interiors). This induces a partitioning of the coefficient matrix A as

$$A = \begin{pmatrix} A_{ss} & A_{sb} & A_{sc} \\ A_{bs} & A_{bb} & A_{bc} \\ A_{cs} & A_{cb} & A_{cc} \end{pmatrix}. \tag{2}$$

In the sequel we use this partitioning also for other matrices of the same order and (unless specified otherwise) refer by 'blocks' to the blocks in such a partitioning. We also denote $f = s \cup b$ (the set of fine grid points, namely, all variables but the coarse grid ones). This induces another block partitioning of A:

$$A = \begin{pmatrix} A_{ff} & A_{fc} \\ A_{cf} & A_{cc} \end{pmatrix},$$

where

$$A_{ff} = \begin{pmatrix} A_{ss} & A_{sb} \\ A_{bs} & A_{bb} \end{pmatrix}. \tag{3}$$

This partitioning is used for other matrices of the same order as well.

For any set g, let $|g|$ denote its cardinality (the number of elements in g). For any positive integer k, let I_k denote the identity matrix of order k. For any set $g \subset c \cup f$, let $J_g : l_2(c \cup f) \to l_2(g)$ denote the injection operator defined by

$$(J_g w)_j = w_j, \ w \in l_2(c \cup f), \ j \in g.$$

For any matrix M, $M = (m_{i,j})_{1 \le i \le K,\ 1 \le j \le L}$, define the absolute value of M by $|M| = (|m_{i,j}|)_{1 \le i \le K,\ 1 \le j \le L}$ and the diagonal matrix of row-sums of M by

$$rs(M) = diag\left(\sum_{j=1}^{L} m_{i,j}\right)_{1 \le i \le K}.$$

Let us define a matrix $T(A)$ which is obtained from A by 'throwing' certain matrix elements onto the main diagonal. More specifically, $T(A)$ is of the same order as A and is upper block-triangular (with respect to the partitioning (2)) with $T(A)_{cc} = I_{|c|}$. The structure of $T(A)$ is thus

$$T(A) = \begin{pmatrix} T(A)_{ss} & T(A)_{sb} & T(A)_{sc} \\ 0 & T(A)_{bb} & T(A)_{bc} \\ 0 & 0 & I_{|c|} \end{pmatrix}.$$

Furthermore, $T(A)_{bb}$ is block-diagonal, with blocks corresponding to interface or boundary segments. Consider, for example, the unknowns on the thick line in Figure 3. Denote by C the set of the two variables denoted by '•', by B the set of the other

Figure 3 First prolongation step; from C, the set of variables denoted by '•', into B, the set of all the other variables on the thick line.

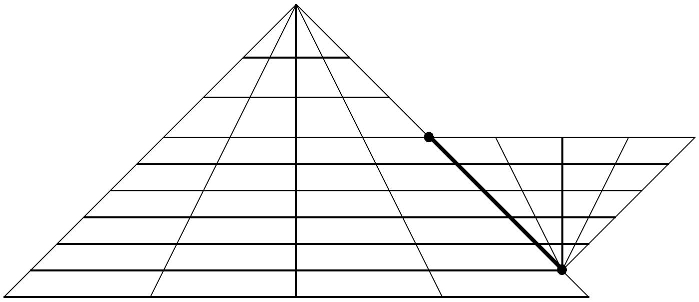
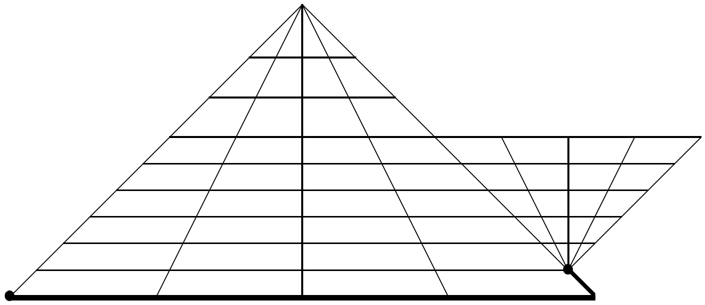


Figure 4 First prolongation step; from C, the set of variables denoted by '•', into B, the set of all the other variables on the thick lines.



variables on the thick line and by F the set of all other variables. The rows in A corresponding to B can be partitioned in the form

$$\left(\begin{array}{ccc} A_{BF} & A_{BB} & A_{BC} \end{array} \right).$$

The corresponding rows in $T(A)$ are defined by

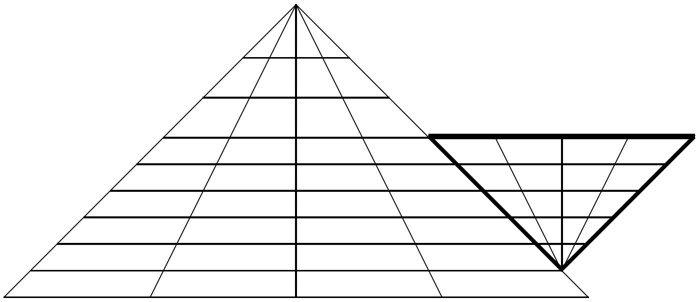
$$\left(\begin{array}{ccc} 0 & A_{BB} - rs(|A_{BF}|) & A_{BC} \end{array} \right).$$

Consequently, the unknowns in B are coupled in $T(A)$ with themselves and with those in C only. It is assumed here that $T(A)_{BB}$ is nonsingular. A sufficient condition for this is that A is diagonally dominant and A_{BB} is irreducible, which implies that $T(A)_{BB}$ is irreducibly diagonally dominant. The rows of $T(A)$ corresponding to other boundary or interface segments of the form $B \subset b$ (such as that of Figure 4) are defined in a similar way. Note that for usual finite element schemes there is no coupling in A between B and $c \setminus C$ and, therefore,

$$T(A)_{bc} = A_{bc}. \tag{4}$$

Similarly, $T(A)_{ss}$ is block diagonal, with blocks corresponding to sets $S \subset s$ corresponding to interiors of subdomains. For example, denote by B the unknowns corresponding to nodes on the thick lines in Figure 5, by S the unknowns corresponding to the nodes in the interior of the subdomain bounded by these lines and by F the rest

Figure 5 Second prolongation step; from B, the set of variables on the thick lines, into S, the set of variables in the interior of the subdomain bounded by the thick lines.



of the unknowns. The rows in A corresponding to S can be partitioned in the form

$$\left(\begin{array}{ccc} A_{SF} & A_{SS} & A_{SB} \end{array} \right).$$

The corresponding rows in $T(A)$ are defined by

$$\left(\begin{array}{ccc} 0 & A_{SS} - rs(|A_{SF}|) & A_{SB} \end{array} \right).$$

Consequently, the unknowns in S are coupled in $T(A)$ with themselves and with those in B only. Rows of $T(A)$ corresponding to other subdomain interiors are defined in a similar way. It is assumed hereafter that $T(A)_{ss}$ is nonsingular. For usual finite element schemes (such as a linear finite element scheme on the mesh in Figure 1) $A_{SF} \equiv 0$ and, therefore,

$$T(A)_{ss} = A_{ss}, \quad T(A)_{sb} = A_{sb} \quad \text{and} \quad T(A)_{sc} = A_{sc}. \tag{5}$$

Therefore, nonsingularity of $T(A)_{ss}$ is guaranteed whenever the differential operator is symmetric positive definite (SPD). (The above more general definition of $T(A)_{ss}$ is introduced for cases where interiors of subdomains might be coupled with each other in A, e.g., when subdomains are not aligned with finite elements. It guarantees that such a coupling cannot exist in $T(A)$, which allows efficient and parallelizable restriction and prolongation operations.)

Define the prolongation operator P and the restriction operator R by

$$P = T(A)^{-1} \quad \text{and} \quad R = P^*,$$

where '$*$' denotes the conjugate with respect to the usual inner product in $l_2(c \cup f)$. Since $T(A)$ is block triangular, the application of P and R is performed easily by block back substitution and block forward elimination, respectively. Furthermore, since the subdomain interiors (such as S in Figure 5) are decoupled from each other in $T(A)$ and the boundary segments (such as B in Figure 3) are also decoupled from each other in $T(A)$, the applications of R and P are highly parallelizable.

Finally, the coarse grid operator Q is a matrix of the same order as A defined by

$$Q = \left(\begin{array}{cc} W & 0 \\ 0 & J_c R A P J_c^t \end{array} \right), \tag{6}$$

where W is a nonsingular matrix of order $|f|$. The reasonable choices for W are

$$W = I_{|f|}$$

or, in the spirit of [Den82],

$$W = R_{ff} diag(A_{ff}) blockdiag(P_{ff}) \tag{7}$$

(where '*blockdiag*' corresponds to the partitioning (3)). The choice (7) yields better numerical results in [Sha97]. It is assumed hereafter that $J_c RAPJ_c^t$ is nonsingular; this is guaranteed, e.g., when A is SPD.

The two-level iteration for the solution of (1) is defined by

$$x_{out} = x_{in} + PQ^{-1}R(\mathbf{b} - Ax_{in}). \tag{8}$$

(8) may be supplemented with relaxations before and after it in the spirit of multigrid. (This approach is used in [Sha96b] for uniform grids and in [Sha97] for locally refined grids.) Alternatively, a Lanczos type acceleration may be applied to it. (This approach is used in [Sha97] for uniform grids.) For both approaches, the condition number of the preconditioned matrix $PQ^{-1}RA$ is an important measure for the rate of convergence. In the following, an upper bound on this condition number is given for a class of SPD problems.

3 Analysis in the SPD Case

Here $(\cdot,\cdot)$ denotes the usual inner product in $l_2(c \cup f)$ and $\|\cdot\|$ denotes the corresponding vector and matrix norms. The terms 'symmetry', 'SPD' and 'orthogonality' used below are interpreted with respect to this inner product. The following lemma is used in the proof of Theorem 1.

Lemma 1 *Let M be a symmetric and positive semidefinite matrix of the same order as A. Then, for any vector $x \in l_2(c \cup f)$,*

$$(x, Mx) \le 2(x, \left(J_f^t J_f M J_f^t J_f + J_c^t J_c M J_c^t J_c\right) x).$$

Proof: Let $\tilde{x} = J_f^t J_f x - J_c^t J_c x$. Then we have

$$0 \le (\tilde{x}, M\tilde{x}) = (x, (J_f^t J_f M J_f^t J_f + J_c^t J_c M J_c^t J_c)x) - (x, (J_f^t J_f M J_c^t J_c + J_c^t J_c M J_f^t J_f)x).$$

The lemma follows from

$$\begin{aligned} (x, Mx) &= (x, (J_f^t J_f M J_f^t J_f + J_c^t J_c M J_c^t J_c)x) + (x, (J_f^t J_f M J_c^t J_c + J_c^t J_c M J_f^t J_f)x) \\ &\le 2(x, (J_f^t J_f M J_f^t J_f + J_c^t J_c M J_c^t J_c)x). \end{aligned}$$

For any matrix M which is SPD with respect to some inner product, define its condition number by

$$\kappa(M) = \rho(M)\rho(M^{-1}),$$

where ρ stands for the spectral radius of a matrix.

Theorem 1 *Assume that A is symmetric and diagonally dominant, $T(A)$ is nonsingular and W is SPD. Then*

$$\begin{aligned}&\kappa(PQ^{-1}RA)\\ &\leq\ 2\max\left\{\rho(W^{-1}J_fRAPJ_f^t),1\right\}\left(1+2\|P_{ff}\|\sqrt{\eta}\|A\|+\eta\|RAP\|+\eta\|W\|\right),\end{aligned}\quad(9)$$

with $\eta=\left(\sqrt{2}+1\right)^2\|A\|$.

Proof: Since A is symmetric and diagonally dominant, it follows from Gershgorin's theorem that it is positive semidefinite. Let $x\in l_2(c\cup f)$ satisfy $\|x\|=1$ and denote $\varepsilon=(x,Ax)$. Since A is symmetric and positive semidefinite, x may be written as a linear combination of the orthogonal eigenvectors of A. Consequently, $\|Ax\|^2\leq\|A\|\varepsilon$.

Define

$$A_1=\begin{pmatrix}(A-T(A))_{ss} & (A-T(A))_{sb} & (A-T(A))_{sc}\\ (A-T(A)^*)_{bs} & rs(|(A-T(A)^*)_{bs}|) & 0\\ (A-T(A)^*)_{cs} & 0 & rs(|(A-T(A)^*)_{cs}|)\end{pmatrix}$$

and

$$A_2=\begin{pmatrix}rs(|A_{sb}|) & A_{sb} & 0\\ A_{bs} & (A-T(A))_{bb} & (A-T(A))_{bc}\\ 0 & (A-T(A)^*)_{cb} & rs(|(A-T(A)^*)_{cb}|)\end{pmatrix}.$$

Note that, when (4) and (5) hold, this simplifies to read

$$A_1\equiv 0\quad\text{and}\quad A_2=\begin{pmatrix}rs(|A_{sb}|) & A_{sb} & 0\\ A_{bs} & (A-T(A))_{bb} & 0\\ 0 & 0 & 0\end{pmatrix}.$$

Since A_1, A_2, $A-A_1$ and $A-A_2$ are symmetric and diagonally dominant, it follows from Gershgorin's theorem that they are positive semidefinite. Using the same argument as in the beginning of the proof, one obtains

$$\|A_nx\|^2\leq\|A_n\|(x,A_nx)\leq\|A_n\|\varepsilon,\quad n=1,2.$$

For convenience we use here the notation $f_1=s$ and $f_2=b$. Note that

$$J_{f_n}A_n=J_{f_n}(A-P^{-1}),\quad n=1,2.$$

Consequently,

$$\begin{aligned}|\ \|J_fAx\|-\|J_fP^{-1}x\|\ |^2 &\leq \|J_f(A-P^{-1})x\|^2=\sum_{n=1}^{2}\|J_{f_n}A_nx\|^2\\ &\leq \sum_{n=1}^{2}\|A_nx\|^2\leq(\|A_1\|+\|A_2\|)\,\varepsilon,\end{aligned}$$

which implies that

$$\|J_fP^{-1}x\|\leq\sqrt{\eta\varepsilon},\quad\text{where }\ \eta=\left(\sqrt{\|A_1\|+\|A_2\|}+\sqrt{\|A\|}\right)^2\leq\left(\sqrt{2}+1\right)^2\|A\|.$$

As a result, we have

$$\begin{aligned}(x, R^{-1}QP^{-1}x) &= (P^{-1}x, QP^{-1}x)\\ &= \left(J_c^tJ_cx + J_f^tJ_fP^{-1}x, Q\left(J_c^tJ_cx + J_f^tJ_fP^{-1}x\right)\right)\\ &\leq (J_c^tJ_cx, RAP(J_c^tJ_cx)) + \eta\|W\|\varepsilon\\ &= \left(P\left(P^{-1}x - J_f^tJ_fP^{-1}x\right), AP\left(P^{-1}x - J_f^tJ_fP^{-1}x\right)\right) + \eta\|W\|\varepsilon\\ &\leq \left(1 + 2\|P_{ff}\|\sqrt{\eta\|A\|} + \eta\|RAP\| + \eta\|W\|\right)\varepsilon,\end{aligned}$$

which implies that the function $(x, R^{-1}QP^{-1}x)/(x, Ax)$ is bounded. On the other hand, we have from Lemma 1 that, for any $y \in l_2(c \cup f)$,

$$\begin{aligned}(y, RAPy) &\leq 2(y, \left(J_f^tJ_fRAPJ_f^tJ_f + J_c^tJ_cRAPJ_c^tJ_c\right)y)\\ &\leq 2\max\left\{\rho(W^{-1}J_fRAPJ_f^t), 1\right\}(y, Qy),\end{aligned}$$

which implies that the function $(x, Ax)/(x, R^{-1}QP^{-1}x)$ is bounded. This completes the proof of the theorem.

Although W in (7) is not SPD, Theorem 1 can be applied with W replaced by an SPD matrix such as $blockdiag(W)$ (or $diag(W)$, or $I_{|f|}$). Using this, Theorem 1 applies also to the nonsymmetric matrix W of (7), provided that the bound in (9) is multiplied by the condition number of

$$P\begin{pmatrix} blockdiag(W)^{-1}W & 0\\ 0 & I_{|c|}\end{pmatrix}P^{-1} \quad \text{(see [Sha96b] for the details).}$$

Note that the above analysis would still be valid if

1. the inner product used above had been replaced by the inner product $(D\cdot, \cdot)$ induced by some diagonal SPD matrix D;
2. the above vector norm had been replaced by $\|\cdot\|_D = \sqrt{(D\cdot, \cdot)}$;
3. the above matrix norm had been replaced by the matrix norm induced by D, that is, $\|\cdot\|_D = \|D^{1/2}\cdot D^{-1/2}\|$
4. and the terms 'symmetry', 'SPD' and 'orthogonality' and the conjugate '*' had been interpreted with respect to $(D\cdot, \cdot)$.

Consequently, we have

Corollary 1 *Assume that A is symmetric and diagonally dominant and $T(D^{-1}A)$ is nonsingular, where D is a diagonal SPD matrix. Let $\tilde{T}(A)$ be the same matrix as $T(A)$ but with the lower right block $I_{|c|}$ replaced by D_{cc}. Define*

$$P = T(D^{-1}A)^{-1} = \tilde{T}(A)^{-1}D, \qquad R = D^{-1}P^tD = \tilde{T}(A)^{-t}D$$
$$and \qquad Q = blockdiag(I_{|f|}, J_cRAPJ_c^t).$$

Then

$$\begin{aligned}&\kappa(PQ^{-1}RD^{-1}A)\\ &\leq 2\max\left\{\rho(J_fRD^{-1}APJ_f^t), 1\right\}\left(1 + 2\|P_{ff}\|_{D_{ff}}\sqrt{\eta\|D^{-1}A\|_D} + \eta\|RD^{-1}AP\|_D + \eta\right)\\ &= 2\max\left\{\rho(J_f\tilde{R}\tilde{A}\tilde{P}J_f^t), 1\right\}\left(1 + 2\|J_f\tilde{P}J_f^t\|\sqrt{\eta\|\tilde{A}\|} + \eta\|\tilde{R}\tilde{A}\tilde{P}\| + \eta\right), \qquad (10)\end{aligned}$$

where $\tilde{A} = D^{-1/2}AD^{-1/2}$, $\tilde{P} = D^{1/2}\tilde{T}(A)^{-1}D^{1/2}$ *and* $\tilde{R} = D^{1/2}\tilde{T}(A)^{-t}D^{1/2}$

$$\textit{and } \eta = \left(\sqrt{2}+1\right)^2 \|D^{-1}A\|_D = \left(\sqrt{2}+1\right)^2 \|\tilde{A}\|.$$

4 Discussion

The bound in (10) depends on $\|\tilde{A}\|$, $\|\tilde{R}\|$ and $\|\tilde{P}\|$. The latter two quantities may be large if a submatrix of the form $T(A)_{BB}$ (corresponding to B in Figures 3 or 4) is nearly singular, which might happen if the variables in B are nearly decoupled in A, or if a submatrix of the form A_{SS} (corresponding to S in Figure 5) is nearly singular, which might happen if $|S|$ is large. Consequently, we have

Corollary 2 *Assume that*

1. *A well-posed problem of the form* $(\mathcal{D}u_x)_x + (\mathcal{D}u_y)_y = f$ *(with suitable boundary conditions and* $\mathcal{D}$ *a uniformly positive function) is considered.*
2. *The scheme is a linear finite element scheme on a triangle mesh.*
3. *The element angles are all less than or equal to* $\pi/2$.
4. *The element angles of the form* $\angle XYZ$ *where the nodes* X *and* Z *lie on one boundary or interface segment are all less than or equal to a mesh-independent constant smaller than* $\pi/2$.
5. *The number of elements per subdomain is bounded mesh-independently.*
6. $D = diag(A)$ *is defined and then (1) is scaled in advance by multiplying it from the left by* D^{-1} *and then the operators* $P = T(A)^{-1}$, $R = D^{-1}P^tD$ *and* Q *as in (6) are defined.*

Then ADD converges with a convergence rate which is independent of both the mesh-size and the jump in the diffusion coefficient $\mathcal{D}$.

Indeed, since (1) is scaled in advance, the coefficient matrix is symmetric with respect to $(D\cdot,\cdot)$ and, therefore, Corollary 1 implies that the convergence rate is independent of the jump in the diffusion coefficient $\mathcal{D}$. Using the other assumptions, it follows from Section 4 in [Sha97] that A is diagonally dominant and that the matrices of the form $T(A)_{BB}$ and $T(A)_{SS}$ are diagonally dominant M-matrices of a bounded order with main diagonal elements bounded away from zero and, therefore, their inverses are of a bounded norm.

Since the definition of ADD relies on the domain decomposition and the coefficient matrix only, it is applicable also to non-conformal schemes and nonmatching grids, provided that the underlying linear system is available. Furthermore, it can be extended in a natural way to the case of overlapping subdomains. The key is the definition of the prolongation operator from the coarse grid to the rest of the nodes. The first prolongation step determining the values at the interface and boundary unknowns is done as before. The second prolongation step determining the values in subdomain interiors is also done as before in each subdomain separately, and then the average of the multiply defined values is taken for nodes in the overlapping area.

Acknowledgement This work was performed in part under the auspices of the US Department of Energy.

REFERENCES

[BPS86] Bramble J. H., Pasciak J. E., and Shatz A. H. (1986) The constructuring of preconditioners for elliptic problems on regions partitioned into substructures *ii. Math. Comp.* 47: 103–134.

[Den82] Dendy J. E. (1982) Black box multigrid. *J. Comput. Phys.* 48: 366–386.

[KM81] Kettler R. and Meijerink J. A. (1981) A multigrid method and a combined multigrid-conjugate gradient method for elliptic problems with strongly discontinuous coefficients in general domains. Technical Report 604, KSELP, Rijswijk, The Netherlands.

[Sha94] Shapira Y. (October 1994) Multigrid methods for 3-d definite and indefinite problems (revised version). Technical Report 834, Computer Science Department, Technion, Haifa, Israel.

[Sha96a] Shapira Y. (1996) Multigrid techniques for highly indefinite equations. In Melson N. D., Manteuffel T. A., McCormick S. F., and Douglas C. C. (eds) *Seventh Copper Mountain Conference on Multigrid Methods*, volume CP 3339, pages 689–705. Hampton, VA.

[Sha96b] Shapira Y. (1996) Two-level analysis and multigrid methods for spd, non-normal and indefinite problems. *SIAM J. Sci. Comput. (submitted)* .

[Sha97] Shapira Y. (1997) Multigrid for locally refined meshes. *SIAM J. Sci. Comput. (submitted)* .

25

Preconditioning Discrete Approximations of the Reissner–Mindlin Plate Model

Douglas N. Arnold, Richard S. Falk, and Ragnar Winther

1 Introduction

The purpose of this paper is to summarize the work of [AFW97]. We consider iterative methods for the solution of indefinite linear systems of equations arising from discretizations of the Reissner–Mindlin plate model.

Like the biharmonic plate model, the Reissner–Mindlin model is a two-dimensional plate model which approximates the behavior of a thin linearly elastic three-dimensional body using unknowns and equations defined only on the middle surface, Ω, of the plate. The basic variables of the model are the transverse displacement ω and the rotation vector $\boldsymbol{\phi}$ which solve the system of partial differential equations

$$\begin{aligned} -\operatorname{\mathbf{div}} \mathcal{C}\mathcal{E}\boldsymbol{\phi} + \lambda t^{-2}(\boldsymbol{\phi} - \operatorname{\mathbf{grad}} \omega) &= 0, \\ \lambda t^{-2}(-\Delta\omega + \operatorname{div} \boldsymbol{\phi}) &= g, \end{aligned} \tag{1}$$

on Ω together with suitable boundary conditions. For the hard clamped plate, which we consider throughout this paper, these are $\omega = 0$, $\boldsymbol{\phi} = 0$. In (1), g is the scaled transverse loading function, t is the plate thickness, $\mathcal{E}\boldsymbol{\phi}$ is the symmetric part of the gradient of $\boldsymbol{\phi}$, and the scalar constant λ and constant tensor $\mathcal{C}$ depend on the material properties of the body.

A variational formulation of this system states that the solution $(\boldsymbol{\phi}, \omega)$ minimizes the total energy of the plate, which is given by

$$E(\boldsymbol{\phi}, \omega) = \frac{1}{2}\int_\Omega \{(\mathcal{C}\mathcal{E}\boldsymbol{\phi}) : (\mathcal{E}\boldsymbol{\phi}) + \lambda t^{-2}|\boldsymbol{\phi} - \operatorname{\mathbf{grad}} \omega|^2\}dx - \int_\Omega g\omega dx \tag{2}$$

over $\boldsymbol{H}_0^1(\Omega) \times H_0^1(\Omega) = \boldsymbol{H}_0^1 \times H_0^1$. Here $H^1 \subset L^2 = L^2(\Omega)$ denotes the Sobolev space of functions with first derivatives in L^2, while H_0^1 is the subspace of functions which vanish on the boundary. Boldface symbols are used to denote 2–vector valued functions and function spaces.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

An advantage of the Reissner–Mindlin model over the biharmonic plate model is that the energy involves only first derivatives of the unknowns and so conforming finite element approximations require the use of merely continuous finite element spaces rather than the C^1 spaces required for the biharmonic model. However, for many choices of finite element spaces, severe difficulties arise due to the presence of the small parameter t. If the finite element subspaces are not properly related, the phenomenon of "locking" occurs, causing a deterioration in the approximation as the plate thickness t approaches zero. A key step in understanding and overcoming locking is passage to a mixed formulation of the Reissner–Mindlin model. The mixed formulation may be derived from the alternative system of differential equations

$$\begin{aligned} -\,\mathbf{div}\,\mathcal{C}\mathcal{E}\boldsymbol{\phi}-\boldsymbol{\zeta} &= 0,\\ -\operatorname{div}\boldsymbol{\zeta} &= g,\\ -\boldsymbol{\phi}+\mathbf{grad}\,\omega-\lambda^{-1}t^2\boldsymbol{\zeta} &= 0, \end{aligned} \tag{3}$$

arising from (1) through the introduction of the shear stress $\boldsymbol{\zeta}=-\lambda t^{-2}(\boldsymbol{\phi}-\mathbf{grad}\,\omega)$. This mixed system also makes sense for $t=0$. In this case the system corresponds to a constrained minimization problem.

2 Mapping Properties

The system (3) can be written in the form

$$\mathcal{A}_t\begin{pmatrix}\boldsymbol{\phi}\\ \omega\\ \boldsymbol{\zeta}\end{pmatrix}=\begin{pmatrix}0\\ g\\ 0\end{pmatrix}, \tag{4}$$

where the coefficient operator, $\mathcal{A}_t$, is given by

$$\mathcal{A}_t=\begin{pmatrix}-\,\mathbf{div}\,\mathcal{C}\mathcal{E} & 0 & -\boldsymbol{I}\\ 0 & 0 & -\operatorname{div}\\ -\boldsymbol{I} & \mathbf{grad} & -\lambda^{-1}t^2\boldsymbol{I}\end{pmatrix}.$$

The mapping properties of the coefficient operator of the continuous system are key to the design of preconditioners for discrete approximations of the system. The indefinite operator $\mathcal{A}_t$ is L^2–symmetric, and, for any $t>0$ is an isomorphism from $\boldsymbol{H}_0^1\times H_0^1\times\boldsymbol{L}^2$ to the L^2–dual $\boldsymbol{H}^{-1}\times H^{-1}\times\boldsymbol{L}^2$. However, in order to obtain bounds on the operator norms which are independent of the thickness t, we are forced to introduce t–dependent norms. Let

$$\boldsymbol{H}_0(\mathrm{rot})=\{\boldsymbol{\eta}\in\boldsymbol{L}^2:\operatorname{rot}\boldsymbol{\eta}\in\boldsymbol{L}^2,\boldsymbol{\eta}\cdot\boldsymbol{s}=0\quad\text{on }\partial\Omega\},$$

with the natural norm. Here $\boldsymbol{s}$ is the unit tangent to $\partial\Omega$ and $\operatorname{rot}\boldsymbol{\eta}=\partial\boldsymbol{\eta}_1/\partial y-\partial\boldsymbol{\eta}_2/\partial x$.

It can be shown that the dual space of $\boldsymbol{H}_0(\mathrm{rot})$ with respect to the L^2–inner product is given by

$$\boldsymbol{H}^{-1}(\mathrm{div})=\{\boldsymbol{\eta}\in\boldsymbol{H}^{-1}:\operatorname{div}\boldsymbol{\eta}\in H^{-1}\}.$$

Using sums and intersections of Hilbert spaces (cf. [BL76]) we now define

$$X_t=\boldsymbol{H}_0^1\times H_0^1\times[\boldsymbol{H}^{-1}(\mathrm{div})\cap t\cdot\boldsymbol{L}^2]$$

and its L^2–dual

$$X_t^* = \boldsymbol{H}^{-1} \times H^{-1} \times [\boldsymbol{H}_0(\mathrm{rot}) + t^{-1} \cdot \boldsymbol{L}^2].$$

In particular,

$$X_0 = \boldsymbol{H}_0^1 \times H_0^1 \times \boldsymbol{H}^{-1}(\mathrm{div}) \qquad \text{and} \qquad X_0^* = \boldsymbol{H}^{-1} \times H^{-1} \times \boldsymbol{H}_0(\mathrm{rot}).$$

Using these spaces, it is then possible to establish the following result.

Theorem 25.1 *The operator $\mathcal{A}_t$ is an isomorphism from X_t to X_t^*. Furthermore, the associated operator norms $||\mathcal{A}_t||_{\mathcal{L}(X_t,X_t^*)}$ and $||\mathcal{A}_t^{-1}||_{\mathcal{L}(X_t^*,X_t)}$ are independent of t.*

3 Preconditioning

Before turning to the description of discretizations schemes, we will discuss preconditioning for the continuous system (4). Our aim is to replace the system (4) by an equivalent system of the form

$$\mathcal{B}_t\mathcal{A}_t \begin{pmatrix} \phi \\ \omega \\ \zeta \end{pmatrix} = \mathcal{B}_t \begin{pmatrix} 0 \\ g \\ 0 \end{pmatrix}, \tag{5}$$

which is more easily solved by iterative methods. The operator $\mathcal{B}_t$ will be symmetric positive definite and hence the indefinite operator $\mathcal{B}_t\mathcal{A}_t$ will be symmetric with respect to the inner product $(\mathcal{B}_t^{-1}\cdot, \cdot)$ on X_t.

Let $\boldsymbol{D}_t$ denote the operator

$$\boldsymbol{D}_t = \boldsymbol{I} + (1 - t^2)\,\mathbf{curl}(I - t^2\Delta)^{-1}\,\mathrm{rot},$$

where

$$\mathbf{curl} = \begin{pmatrix} -\partial/\partial y \\ \partial/\partial x \end{pmatrix}.$$

When $t = 0$, this is a differential operator which is an isomorphism from $\boldsymbol{H}_0(\mathrm{rot})$ into $\boldsymbol{H}^{-1}(\mathrm{div})$. In general, it can be shown that that $\boldsymbol{D}_t$ is an isomorphism from $\boldsymbol{H}_0(\mathrm{rot}) + t^{-1} \cdot \boldsymbol{L}^2$ to $\boldsymbol{H}^{-1}(\mathrm{div}) \cap t \cdot \boldsymbol{L}^2$, with the operator norms of $\boldsymbol{D}_t$ and $\boldsymbol{D}_t^{-1}$ independent of t. An immediate consequence of this is that the block diagonal operator

$$\mathcal{B}_t = \begin{pmatrix} -\boldsymbol{\Delta}^{-1} & 0 & 0 \\ 0 & -\Delta^{-1} & 0 \\ 0 & 0 & \boldsymbol{D}_t \end{pmatrix}$$

is an isomorphism mapping X_t^* to X_t with the norms $||\mathcal{B}_t||_{\mathcal{L}(X_t^*,X_t)}$ and $||\mathcal{B}_t^{-1}||_{\mathcal{L}(X_t,X_t^*)}$ independent of t. From Theorem 25.1, we conclude that the block diagonal positive definite operator $\mathcal{B}_t$ has the same mapping property as $\mathcal{A}_t^{-1}$. Hence, the composition $\mathcal{B}_t\mathcal{A}_t$,

$$X_t \xrightarrow{\mathcal{A}_t} X_t^* \xrightarrow{\mathcal{B}_t} X_t,$$

is an isomorphism from X_t to X_t with operator norms

$$||\mathcal{B}_t\mathcal{A}_t||_{\mathcal{L}(X_t,X_t)} \quad \text{and} \quad ||(\mathcal{B}_t\mathcal{A}_t)^{-1}||_{\mathcal{L}(X_t,X_t)}$$

independent of t. Therefore, $\mathcal{B}_t\mathcal{A}_t$ is a bounded operator on X_t with bounded inverse, and as a consequence, the spectral condition number

$$\kappa(\mathcal{B}_t\mathcal{A}_t) = \frac{\sup|\sigma(\mathcal{B}_t\mathcal{A}_t)|}{\inf|\sigma(\mathcal{B}_t\mathcal{A}_t)|}$$

is finite and independent of t.

A preconditioned differential system of the form (5) can be solved by a Krylov space method like MINRES or CGN (conjugate gradients applied to the normal equations). These methods are well defined as long as the coefficient operator $\mathcal{B}_t\mathcal{A}_t$ maps X_t into itself, and convergence in the norm of X_t is guaranteed as long as the spectral condition number of $\mathcal{B}_t\mathcal{A}_t$ is finite. Therefore, we obtain the following result.

Theorem 25.2 *Assume that MINRES or CGN is applied to the preconditioned system* (5). *Then the sequence of approximations converges to the solution in* X_t, *with a convergence rate independent of* t.

4 Stable Discretizations

The continuous theory presented above may serve as a guideline for the problem of real interest, i.e., how to construct effective preconditioners for the discrete systems arising from finite element approximations of the differential system (3). Here, we shall just give a brief outline of the discrete theory. For full details and proofs we refer to the original paper [AFW97].

A finite element approximation of the system (3) (or (4)) will typically give rise to a discrete system of the form

$$\mathcal{A}_{t,h}\begin{pmatrix}\phi_h\\ \omega_h\\ \zeta_h\end{pmatrix} = \begin{pmatrix}0\\ g_h\\ 0\end{pmatrix},$$

where $\mathcal{A}_{t,h}$ is an indefinite, L^2–symmetric operator mapping a finite dimensional space $X_h = \boldsymbol{V}_h \times W_h \times \boldsymbol{\Gamma}_h$ into itself. Here $h > 0$ is a discretization parameter. Examples of stable and locking free finite element discretizations have been proposed by Arnold and Falk [AF89], Brezzi, Fortin and Stenberg [BFS91], Duran and Liberman [DL92], and others. The main purpose of this work is to construct preconditioners $\mathcal{B}_{t,h}$ for these systems such that the spectral condition number of $\mathcal{B}_{t,h}\mathcal{A}_{t,h}$ is independent of the thickness t and the discretization parameter h. The construction of $\boldsymbol{D}_{t,h}$ is analogous to that of $\boldsymbol{D}_t$ in the continuous case.

For the locking free methods, it is possible to establish a discrete version of Theorem 25.1 above, i.e., to prove that the operator norms

$$||\mathcal{A}_{t,h}||_{\mathcal{L}(X_{t,h},X^*_{t,h})} \quad \text{and} \quad ||\mathcal{A}^{-1}_{t,h}||_{\mathcal{L}(X^*_{t,h},X_{t,h})}$$

are bounded uniformly in t and h. Here the spaces $X_{t,h}$ and $X^*_{t,h}$ coincide with X_h as a set, but are endowed with norms which resemble the norms in X_t and X^*_t. As in the continuous case, this property of $\mathcal{A}_{t,h}$ suggests a symmetric, positive definite and block diagonal preconditioner such that

$$||\mathcal{B}_t||_{\mathcal{L}(X^*_{t,h},X_{t,h})} \quad \text{and} \quad ||\mathcal{B}_t^{-1}||_{\mathcal{L}(X_{t,h},X^*_{t,h})}$$

are independent of t and h. As a consequence, the spectral condition number of $\mathcal{B}_{t,h}\mathcal{A}_{t,h}$ is independent of t and h.

The preconditioner $\mathcal{B}_{t,h}$ will be of the form

$$\mathcal{B}_t = \begin{pmatrix} \boldsymbol{L}_h & 0 & 0 \\ 0 & M_h & 0 \\ 0 & 0 & \boldsymbol{N}_{t,h} \end{pmatrix},$$

where $\boldsymbol{L}_h$ and M_h are chosen spectrally equivalent to approximations of the inverse of the negative Laplace operator on $\boldsymbol{V}_h$ and W_h, respectively, while $\boldsymbol{N}_{t,h}$ is a discrete analog of $\boldsymbol{D}_t$.

5 Numerical Examples

In the examples presented below, the domain Ω is taken to be the unit square. A triangulation of Ω is obtained by first dividing Ω into squares of size $h \times h$, and then dividing each square into two triangles using the positively sloped diagonal. All the computations are done with the method of Arnold and Falk [AF89]. Hence, the space $\boldsymbol{V}_h$ consists of continuous piecewise linear functions plus cubic bubbles on each triangle, W_h is the nonconforming piecewise linear space, with continuity requirements only at the midpoint of each edge, and $\boldsymbol{\Gamma}_h$ is the space of piecewise constants with respect to the triangulation.

The operators $\boldsymbol{L}_h$ and M_h are essentially constructed from a standard V–cycle multigrid operator with a Gauss–Seidel smoother. These operators are fixed throughout all the experiments. For the method considered here, the proper discrete analog of the operator $\boldsymbol{D}_t$ is of the form

$$\boldsymbol{I} + (1 - t^2)\,\mathbf{curl}_h(I - t^2\Delta_h)^{-1}\,\mathrm{rot}_h$$

mapping $\boldsymbol{\Gamma}_h$ into itself. If $Q_h \subset H^1$ is the space of continuous piecewise linear functions with respect to the triangulation, then $\mathbf{curl}_h : Q_h \mapsto \boldsymbol{\Gamma}_h$ is defined by restricting the ordinary $\mathbf{curl}$–operator to Q_h. Furthermore, $\mathrm{rot}_h : \boldsymbol{\Gamma}_h \mapsto Q_h$ is the adjoint operator, while $\Delta_h : Q_h \mapsto Q_h$ is the standard finite element approximation of the Laplace operator generated by the space Q_h.

By replacing $(I - t^2\Delta_h)^{-1}$ by a spectrally equivalent (with respect to t and h) preconditioner $\Phi_{t,h}$, again derived from a V–cycle multigrid algorithm, we obtain a computational feasible operator

$$\boldsymbol{D}_{t,h} = \boldsymbol{I} + (1 - t^2)\,\mathbf{curl}_h\,\Phi_{t,h}\,\mathrm{rot}_h\,.$$

We observe that the operator $\boldsymbol{D}_{t,h}$ simplifies when $t = 0$, by taking $\Phi_{0,h} = I$, and for $t = 1$, since $\boldsymbol{D}_{1,h} = \boldsymbol{I}$.

In the examples below, the preconditioned system is solved either by MINRES or CGN. The work estimate for one iteration of CGN corresponds roughly to two MINRES iterations. We therefore compare the number of iterations for MINRES (N_{MR}) with twice the number of iterations for CGN (N_{CGN}). The condition number $\kappa(\mathcal{B}_{t,h}\mathcal{A}_{t,h})$, which is estimated from the conjugate gradient iteration using a standard

Matlab routine, will also be given. The iterations are terminated when the error, measured in the norm associated with the inner product $(\mathcal{B}_{t,h}^{-1}\cdot, \cdot)$, is reduced by a factor of at least $5 \cdot 10^4$.

The two extreme cases $t = 0$ and $t = 1$ are considered in Tables 1 and 2.

Table 1 $t = 0$, $\boldsymbol{N}_{t,h} = \boldsymbol{D}_{t,h} = \boldsymbol{D}_{0,h}$

h	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
N_{MR}	41	41	35	29	24
N_{CGN}	48	50	48	40	34
$\kappa(\mathcal{B}_{t,h}\mathcal{A}_{t,h})$	8.17	10.7	11.1	10.6	9.62

Table 2 $t = 1$, $\boldsymbol{N}_{t,h} = \boldsymbol{D}_{t,h} = \boldsymbol{I}$

h	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
N_{MR}	22	22	20	20	20
N_{CGN}	102	104	106	104	102
$\kappa(\mathcal{B}_{t,h}\mathcal{A}_{t,h})$	17.5	18.4	19.0	19.0	18.9

Table 3 $\kappa(\mathcal{B}_{t,h}\mathcal{A}_{t,h})$ for $t = 0.01$

h	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
$\boldsymbol{N}_{t,h} = \boldsymbol{D}_{0,h}$	8.15	10.7	11.4	32.9	113
$\boldsymbol{N}_{t,h} = \boldsymbol{D}_{t,h}$	8.15	10.7	11.2	11.1	9.68

The results clearly seem to confirm the boundedness of $\kappa(\mathcal{B}_{t,h}\mathcal{A}_{t,h})$ with respect to h for these values of t. Observe also the substantial difference in the behavior of MINRES and CGN in the case $t = 1$.

Our theory predicts that if $\boldsymbol{N}_{t,h}$ is chosen to be $\boldsymbol{D}_{t,h}$, then the condition numbers $\kappa(\mathcal{B}_{t,h}\mathcal{A}_{t,h})$ (and hence N_{MR} and N_{CGN}) are bounded independently of t and h. Furthermore, if t is sufficiently small compared to h ($t = O(h)$) then the simpler choice $\boldsymbol{N}_{t,h} = \boldsymbol{D}_{0,h}$ is a good one as well. In Table 3, we compare the condition numbers $\kappa(\mathcal{B}_{t,h}\mathcal{A}_{t,h})$ for $t = 0.01$ and $\boldsymbol{N}_{t,h} = \boldsymbol{D}_{t,h}$ or $\boldsymbol{N}_{t,h} = \boldsymbol{D}_{0,h}$. As expected, the choice $\boldsymbol{N}_{t,h} = \boldsymbol{D}_{0,h}$ works well when h is large, but deteriorates when h becomes too small. In contrast, the condition numbers for the choice $\boldsymbol{N}_{t,h} = \boldsymbol{D}_{t,h}$ appear to be bounded uniformly with respect to h.

For any fixed $t > 0$ the choice $\boldsymbol{N}_{t,h} = \boldsymbol{D}_{1,h} = \boldsymbol{I}$ leads to condition numbers $\kappa(\mathcal{B}_{t,h}\mathcal{A}_{t,h})$ which are independent of h, but which may increase with decreasing values of t. The effect of this is illustrated in Table 4. These results confirm the uniformity with respect to h for the simple choice $\boldsymbol{N}_{t,h} = \boldsymbol{I}$, but the experiments also clearly indicate that this is not a good choice for t sufficiently small.

Table 4 $\kappa(\mathcal{B}_{t,h}\mathcal{A}_{t,h})$ for $t = 0.1$

h	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}
$\boldsymbol{N}_{t,h} = \boldsymbol{I}$	90.2	78.5	72.7	70.1	70.7
$\boldsymbol{N}_{t,h} = \boldsymbol{D}_{t,h}$	8.64	10.8	11.1	11.2	11.1

Acknowledgement

The first author was supported by NSF grants DMS-9205300 and DMS-9500672 and by the Institute for Mathematics and its Applications. The second author was supported by NSF grants DMS-9403552. The third author was supported by The Norwegian Research Council under grants 100331/431 and STP.29643.

REFERENCES

[AF89] Arnold D. N. and Falk R. S. (1989) A uniformly accurate finite element method for the Reissner–Mindlin plate. *SIAM J. Numer. Anal.* 26: 1276–1290.

[AFW97] Arnold D. N., Falk R. S., and Winther R. (1997) Preconditioning discrete approximations of the Reissner–Mindlin plate model. *to appear in Math. Mod. Num. Anal.* .

[BFS91] Brezzi F., Fortin M., and Stenberg R. (1991) Error analysis of mixed-interpolated elements for Reissner–Mindlin plates. *Math. Models and Methods in Applied Sciences* 1: 125–151.

[BL76] Bergh J. and Löfstrom J. (1976) *Interpolation spaces, an introduction.* Springer Verlag.

[DL92] Durán R. and Liberman E. (1992) On mixed finite element methods for the Reissner–Mindlin plate model. *Math. Comp.* 58: 561–573.

26

A Non–overlapping Domain Decomposition Method for Solving Elliptic Problems by Finite Element Methods

Xiaobeng Feng

1 Introduction

Non-overlapping domain decomposition methods have received a lot attention during the last few years, due to the restrictions of overlapping domain decomposition methods. Several families of non–overlapping decomposition methods for the solutions of elliptic problems have been proposed, analyzed, and successfully implemented [BW86, BPWX91, DPLRW93, Dry89, GW88, Lio90, MQ89, LTDRV91, Tan92].

In a non–overlapping domain decomposition method, the original problem is first decomposed into smaller problems defined on non–overlapping subdomains. Parallel or sequential iterative procedures are then constructed for decoupling the whole domain problem into subdomain problems. During the iterative process, information must be transmitted between subdomains in order to guarantee convergence. This "information transmission" step is the key part of a domain decomposition method; it distinguishes one domain decomposition method from another. Several methods for passing information have been proposed in the literature [BF96, DPLRW93, GLT90, Lio90, MQ89]. The common approach was to develop a transmission condition for the differential problem and adapt the same condition to the corresponding discrete problem.

The purpose of this paper is to present a parallelizable, iterative, non–overlapping domain decomposition method for solving second oreder elliptic problems discretized by finite element methods. Unlike the usual approach, we bypass the differential problems and construct the iterative procedure based on domain decomposition techniques directly for the finite element equations. To obtain the split subdomain problems, our main idea is to use a penalty method on each subdomain and to introduce a local (in the pointwise sense), non–Robin type transmission condition which not only enhances the convergence and passes the information between the

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

subdomains but also traces the jumps of the discrete Neumann data of the finite element solutions across the interfaces.

The rest of the paper is organized as follows. In Section 2, a model second order elliptic problem and its finite element discretization are introduced. In Section 3, a parallelizable iterative procedure based on domain decomposition techniques for solving the finite element equations of the elliptic problem is presented. Finally, in section 4, the convergence analysis and the rate of convergence are demonstrated in the case when subdomains are chosen as small as the individual finite elements to show the effectiveness of the procedure. It is shown that the domain decomposition procedure converges at a rate which is independent of the mesh size h if the relaxation parameters are chosen properly. All the theorems and lemmas are stated either without proofs or with schematic proofs. For the details, we refer to [Fen96], where numerical experiments are also presented, and closely related domain decomposition procedures are developed for the biharmonic equation and the Helmholtz equation.

2 The Model Problem

Let $\Omega \subset \mathbf{R}^2$ be a bounded polygonal domain. Consider the model Dirichlet problem:

$$\begin{aligned} -\Delta u(x) + c(x)u(x) = &\quad f(x), \qquad \text{in } \Omega, &(1)\\ u(x) = &\quad 0, \qquad \text{on } \partial\Omega, &(2)\end{aligned}$$

where the coefficient function $c(x) \geq 0$. The weak formulation of (1)–(2) is to find $u \in H_0^1(\Omega)$ such that

$$a(u,v)_\Omega = (f,v)_\Omega, \qquad \forall v \in H_0^1(\Omega), \tag{3}$$

where

$$a(w,v)_\Omega = \int_\Omega \nabla w \cdot \nabla v \; dx, \qquad (w,v)_\Omega = \int_\Omega wv dx.$$

Let $\mathcal{T}_h$ be a quasiuniform triangular or rectangular partition of Ω and $V^h \subset H_0^1(\Omega)$ denote a finite element space of piecewise polynomials of degree r (≥ 1). Then the finite element method for problem (1)–(2) is to find $u^h \in V^h$ such that

$$a(u^h,v)_\Omega = (f,v)_\Omega, \qquad \forall v \in V^h. \tag{4}$$

Let $\{\phi_j^h\}_{j=1}^n$ denote the nodal basis of the finite element space V^h and $\{p_j^h\}_{j=1}^n$ denote the nodal set corresponding to the nodal parameters (a node is counted k times if there are k nodal parameters attached to it). Then (4) gives the following linear system:

$$A\xi = b, \tag{5}$$

where

$$A = [a(\phi_i^h, \phi_j^h)]_{n\times n}, \quad \xi = (u^h(p_1^h), \cdots, u^h(p_n^h))^t, \quad b = ((f, \phi_1^h)_\Omega, \cdots, (f, \phi_n^h)_\Omega)^t.$$

It is well–known that the condition number of the system is of order h^{-2}, which implies that the system is ill–conditioned. Therefore, to solve problem (4), in particular for small h, a fast solver other than a classical iterative method is necessary. To find a required fast solver using domain decomposition techniques is the goal of this paper.

The following notations are adopted throughout the rest of this paper.

Let $\{\Omega_j\}_1^J$ be a non–overlapping partition of Ω, that is,

$$\overline{\Omega} = \cup_{j=1}^J \overline{\Omega}_j; \qquad \Omega_j \cap \Omega_k = \emptyset, \quad \text{if } j \neq k.$$

Assume that $\partial\Omega_j$, $j = 1, 2, \cdots, J$ is Lipschitz and Ω_j is a star–shaped domain. We also assume that the non-overlapping partition aligns with the triangulation $\mathcal{T}_h$. In practice, with the exception of perhaps a few Ω_j's along $\partial\Omega$, each Ω_j will be convex with a piecewise–smooth boundary. For example, an interesting choice for the domain decomposition of a finite element discretization is to let each finite element be a subdomain.

Finally, define

$$\Gamma = \partial\Omega, \qquad \Gamma_j = \Gamma \cap \partial\Omega_j, \qquad \Gamma_{jk} = \Gamma_{kj} = \partial\Omega_j \cap \partial\Omega_k. \tag{6}$$

$$H_{\Gamma_j}^1(\Omega_j) = \{v \in H^1(\Omega_j);\ v = 0, \text{ on } \Gamma_j\}. \tag{7}$$

3 The Domain Decomposition Iterative Method

The objective of this section is to construct a domain decomposition iterative method to solve the finite element equations (4). The key step is to construct the split subdomain problems and the local (in the pointwise sense) transmission conditions on the interfaces of the subdomains. We notice that the pointwise continuity across the element interfaces does not hold for the flux since, in general, $\frac{\partial u_i^h}{\partial n_i}$ is different from $\frac{\partial u_j^h}{\partial n_j}$ on Γ_{ij}. Therefore, any attempt to enforce the pointwise continuity of the flux will not succeed. The above observation leads us to take the following approach: find transmission conditions that preserve the continuity of u^h and trace the *discontinuity* of $\frac{\partial u^h}{\partial n}$ on the interfaces.

To construct the subdomain problems, first, we rewrite (4) as

$$\sum_j a(u^h, v)_{\Omega_j} = \sum_j (f, v)_{\Omega_j}, \qquad \forall v \in V^h. \tag{8}$$

Next, we observe the following fact:

$$\sum_{i,j} \langle \beta(\frac{\partial u_i^h}{\partial n_i} - \frac{\partial u_j^h}{\partial n_j}), v\rangle_{\Gamma_{ij}} = 0, \qquad \forall v \in V^h,$$

for any nonzero constant β.

Now for $V_j^h = V^h|_{\Omega_j}$ and $u_j^h = u^h|_{\Omega_j}$, it is easy to see that $\{u_j^h\}$ satisfies

$$\begin{aligned} a_i(u_i^h, v) - \sum_j \langle \beta(\frac{\partial u_i^h}{\partial n_i} - \frac{\partial u_j^h}{\partial n_j}), v\rangle_{\Gamma_{ij}} &= (f, v)_{\Omega_i}, \ \forall v \in V_i^h, & (9)\\ u_i^h &= u_j^h. \ \text{ on } \Gamma_{ij}. & (10) \end{aligned}$$

We remark that the second term on the right hand side of (9), which measures the total *jumps* of the Neumann data being transmitted into the subdomain Ω_i, can be viewed as a penalty term for the subdomain problem on Ω_i, and the size of the free parameter β strongly influences the size of the penalty term.

On the other hand, it is not convenient to decouple the whole domain problem (4) based on (9)–(10), since the interface condition (10) is a Dirichlet condition on the interfaces for each subdomain problem. To overcome this difficulty, we replace equation (10) by the following equivalent one:

$$-\beta \frac{\partial u_j^h}{\partial n_j} + \alpha u_i^h = -\beta \frac{\partial u_j^h}{\partial n_j} + \alpha u_j^h. \quad \text{on } \Gamma_{ij}, \tag{11}$$

which is obtained by adding $-\beta \frac{\partial u_j^h}{\partial n_j}$ to both sides of (10) after multiplying it by another nonzero constant α.

Remark 3.1 The "new" interface condition (11) still holds in the pointwise sense. This condition is *not* a Robin type transmission condition since the partial derivative on the left hand side is $\frac{\partial u_j^h}{\partial n_j}$ not $\frac{\partial u_i^h}{\partial n_i}$.

Now based on (9)–(10), we propose the following domain decomposition iterative algorithm:

Algorithm 1

Step 1. $\forall u_i^0 \in V_i^h$, $i = 1, 2, \cdots, J$.

Step 2. Compute $\{u_i^n\}$ for $i = 1, 2, \cdots, J$ and $n \geq 1$ by solving

$$a_i(u_i^n, v) - \sum_j \langle \beta \frac{\partial u_i^n}{\partial n_i} + \lambda_{ji}^n, v\rangle_{\Gamma_{ij}} = (f, v)_{\Omega_i}, \quad \forall v \in V_i^h, \tag{12}$$

$$\lambda_{ji}^n + \alpha u_i^n = -\beta \frac{\partial u_j^{n-1}}{\partial n_j} + \alpha u_j^{n-1}, \quad \text{on } \Gamma_{ij}, \tag{13}$$

Note that we have omitted all super indices h in the algorithm.

4 Convergence Analysis

In this section we will establish the convergence of Algorithm 1 and derive an upper bound for its rate of convergence. Our analysis based on the discrete version of an energy method (cf. [DPLRW93]), which was first proposed by Després in [Des91] for analyzing convergence of a Lions' type domain decomposition method for the Helmholtz equation at the differential level.

Let

$$e_i^n = u_i^h - u_i^n, \qquad \mu_{ji}^n = -\beta\frac{\partial u_j^h}{\partial n_j} - \lambda_{ji}^n.$$

Then from (9)–(13) we get

$$a_i(e_i^n, v) - \sum_j \langle \beta\frac{\partial e_i^n}{\partial n_i} + \mu_{ji}^n, v_i\rangle_{\Gamma_{ij}} = 0, \quad \forall v \in V_i^h, \tag{14}$$

$$\mu_{ji}^n + \alpha e_i^n = -\beta\frac{\partial e_j^{n-1}}{\partial n_j} + \alpha e_j^{n-1}, \quad \text{on } \Gamma_{ij}. \tag{15}$$

Clearly,

$$a_i(e_i^n, e_i^n) = \sum_j \langle \beta\frac{\partial e_i^n}{\partial n_i} + \mu_{ji}^n, e_i^n\rangle_{\Gamma_{ij}}. \tag{16}$$

Define the "pseudo–energy"

$$E_n = E(\{e_i^n\}) = \sum_{i,j} |\mu_{ji}^n + \alpha e_i^n|^2_{0,\Gamma_{ij}}. \tag{17}$$

By (13)–(16) we can show the following lemmas (cf. [Fen96]).

Lemma 4.1

$$E_n = E_{n+1} - R_{n-1} = E_0 - \sum_{\ell=0}^{n-1} R_\ell, \tag{18}$$

where

$$R_{n-1} = R(\{e_j^{n-1}\}) = \sum_{i,j}[|\mu_{ij}^{n-1}|^2_{0,\Gamma_{ij}} - \beta^2\left|\frac{\partial e_j^{n-1}}{\partial n_j}\right|^2_{0,\Gamma_{ij}}] + 2\alpha\sum_j a_j(e_j^{n-1}, e_j^{n-1}). \tag{19}$$

Lemma 4.2 *If the parameters α and β are chosen to satisfy $\frac{\alpha}{\beta^2} = O(h^{-1})$, then $R_n \geq 0$ for $n \geq 1$.*

Remark 4.1 The following are sample choices of α and β which satisfy the assumption of Lemma 4.2

1. $\alpha = O(1)$ and $\beta = O(\sqrt{h})$.
2. $\alpha = O(h^{-1})$ and $\beta = O(1)$.
3. $\alpha = O(h)$ and $\beta = O(h)$.

Theorem 4.1 *Choose the parameters α and β such that $\frac{\alpha}{\beta^2} = O(h^{-1})$, then*

1. $\lambda_{ij}^\ell \to -\beta \frac{\partial u_i^h}{\partial n_i}$ in $L^2(\Gamma_{ij})$ as $\ell \to \infty$.
2. $u_j^\ell \to u_j^h$ in $H^1(\Omega_j)$ as $\ell \to \infty$.

Proof. Notice that if $\frac{\alpha}{\beta^2} = O(h^{-1})$, then $\{E_n\}$ is a decreasing sequence. Therefore, if $c(x) \geq C_0 > 0$, the theorem immediately follows from Lemma 4.1 and Lemma 4.2. If $c(x) = 0$ or $c(x) \geq 0$, Lemma 4.1 and Lemma 4.2 imply the convergence of ∇e_i^ℓ in $L^2(\Omega_i)$ for each Ω_i. To show the convergence of e_i^ℓ in $L^2(\Omega_i)$, we first consider all boundary subdomains Ω_j. Since $e_j^\ell = 0$ on Γ_j, by Poincaré inequality we have $e_j^\ell \to 0$ in $L^2(\Omega_j)$ for each boundary subdomain Ω_j. Suppose Ω_i is a subdomain which has a common interface Γ_{ij}^* with one of the boundary subdomains, say, Ω_j. From (15) we have

$$\alpha e_i^\ell = \left(-\frac{\partial e_j^{\ell-1}}{\partial n_j} - \mu_{ji}^\ell \right) + \alpha e_j^{\ell-1}, \quad \text{on } \Gamma_{ij}^*.$$

And

$$\|e_i^\ell\|_{0,\Omega_j} \leq C \left[\|\nabla e_i^\ell\|_{0,\Omega_i} + \int_{\Gamma_{ij}^*} |e_i^\ell|^2 ds \right] \to 0 \qquad \text{as } \ell \to \infty.$$

Hence

$$\|e_i^\ell\|_{H^1(\Omega_j)} \to 0 \qquad \text{as } \ell \to \infty.$$

So the convergence takes place on the subdomain Ω_i. The argument can be repeated until the domain is exhausted. The proof is completed.

The above convergence theorem says that, for appropriately chosen parameters α and β, Algorithm 1 produces a strongly convergent sequence. In the rest of this section we will address the issue of the algorithm's speed of convergence by giving an upper bound estimate for the rate of convergence.

Define

$$T_f(\{u_i^{n-1}\}) = \{u_i^n\}. \tag{20}$$

Then

$$T_f(u) = T_0(u) + T_f(0).$$

If u^* is a fixed point of T_f, then

$$(I - T_0)(u^*) = T_f(0). \tag{21}$$

Lemma 4.3 *Suppose $c(x) \geq C_0 > 0$, and let u be an eigenfunction of T_0. Under the assumption of Theorem 4.1 there exists a constant $Q(h) > 0$ such that*

$$E(u) \leq Q(h)R(u), \qquad \textit{with} \quad Q(h) = 2 + \frac{\alpha h^{-1} C_1}{C_0}, \tag{22}$$

where C_1 is some positive constant which is independent of h.

Remark 4.2 The conclusion of Lemma 4.3 still holds in the case $c(x) \geq 0$. For a detailed proof, see [Fen96].

Theorem 4.2 *Let $\rho(T_0)$ denote the spectral radius of T_0. Then under the assumptions of Lemma 4.3 the following estimate holds:*

$$\rho(T_0) \leq 1 - \frac{1}{Q(h)}, \tag{23}$$

where $Q(h)$ is given in (22).

Proof. Suppose

$$T_0(u) = \gamma u,$$

then from (18) we get

$$|\gamma|^2 E(u) = E(u) - R(u). \tag{24}$$

Hence, the theorem follows from combing Lemma 4.3 and (24)

Remark 4.3 From Theorem 4.2, we immediately conclude that the spectral radius of the iteration matrix of Algorithm 1 has an upper bound of the form $O(h^{-1})$ if $\alpha = O(1)$ and $\beta = O(\sqrt{h})$, moreover, it is bounded by an absolute constant which is less than one if $\alpha = O(h)$ and $\beta = O(h)$, that is, the algorithm converges optimally when $\alpha = O(h)$ and $\beta = O(h)$.

REFERENCES

[BF96] Bennethum L. S. and Feng X. (1996) A domain decomposition method for solving a helmholtz–like problem in elasticity based on the Wilson nonconforming finite element. *R.A.I.R.O. Anal. Numer.* (to appear).

[BPS89] Bramble J. H., Pasciak J. E., and Schatz A. H. (1989) The construction of preconditioners for elliptic problems by substructuring, IV. *Math. Comp.* 53: 1–24.

[BW86] Bjørstad P. and Widlund O. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal.* 23: 1093–1120.

[Des91] Després B. (1991) Domain decomposition method and helmhotz problem. In G. Cohn L. H. and Joly P. (eds) *Proc. SIAM Mathematical and Numerical Aspects of Wave Propagation Phenomena*, pages 44–52. SIAM, Philadelphia.

[DPLRW93] Douglas Jr. J., Paes Leme P. J. S., Roberts J. E., and Wang J. (1993) A parallel iterative procedure applicable to the approximate solution of second order partial differential equations by mixed finite element methods. *Numer. Math.* 65: 95–108.

[DW90] Dryja M. and Widlund O. B. (1990) Towards to a unified theory of domain decomposition algorithms for elliptic problems. In *Proc. Third International Symposium on Domain Decomposition Method for Partial Differential Equations*, pages 53–61. SIAM, Philadelphia.

[Fen96] Feng X. (1996) Parallel iterative domain decomposition method for second and fourth order elliptic problems. preprint.

[GLT90] Glowinski R. and Le Tallec P. (1990) Augmented lagrangian interpretation of the nonoverlapping schwarz alternating method. In *Proc. Third International Symposium on Domain Decomposition Method for Partial Differential Equations.* SIAM, Philadelphia.

[GW88] Glowinski R. and Wheeler M. F. (1988) Domain decomposition and mixed finite element methods for elliptic problems. In *Proc. First International Symposium on Domain Decomposition Method for Partial Differential Equations.* SIAM, Philadelphia.

[Lio90] Lions P. L. (1990) On the schwartz alternating method III: a variant for nonoverlapping subdomains. In *Proc. Third International Symposium on Domain Decomposition Method for Partial Differential Equations.* SIAM, Philadelphia.

[LTDRV91] Le Tallec P., De Roeck Y., and Vidrascu M. (1991) Domain decomposition methods for large linearly elliptic three–dimensional problems. *J. Comp. and Appl. Math.* 34: 93–117.

[MQ89] Marini L. and Quarteroni A. (1989) A relaxation procedure for domain decomposition methods using finite elements. *Numer. Math.* 55: 575–598.

[Tan92] Tang W. P. (1992) Generalized schwarz splitting. *SIAM J. Sci. Stat. Comp* 13: 573–595.

27

A Hybrid Domain Decomposition Method For Convection-Dominated Problems

Magne S. Espedal, Xue-Cheng Tai, and Ningning Yan

1 Introduction

Domain decomposition methods have been intensively studied for partial differential equations. They are efficient parallel methods especially for the elliptic equations. However, when domain decomposition are used for convection-dominated problems, the flow directions must be carefully considered. We refer to [WY97], [TJDE97], [RZ94], [RT97], [KL95] for some results of domain decomposition methods for convection-dominated problems.

In this work a hybrid domain decomposition method is proposed. When the flow is simple, a non-iterative domain decomposition approach can be used. The subdomains in the upwind side shall be computed first and the subdomains in downwind direction are computed one after another. For each subdomain, Dirichlet boundary condition is used on the inflow boundary and an artificial boundary condition is used on the outflow boundary. When the flow is complicated, an iterative method must be used. The proposed methods are suitable for problem (1) when the diffusion parameter ϵ is relatively small. For small ϵ, the error introduced by the domain decomposition methods is small, and one can easily use finer meshes in the subdomains that intersect with singular layers. When the proposed methods are used for time dependent problems, the convergence properties are even better. The proposed methods of this work are easy to implement and easy to do local refinement.

2 The Hybrid Domain Decomposition Method

Consider the advection diffusion problem:

$$\begin{cases} -div(\epsilon \bigtriangledown u) + div(\beta u) + \alpha u = f, & in \ \Omega, \\ u = 0, & on \ \partial\Omega, \end{cases} \tag{1}$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

where α, f are bounded functions, $\boldsymbol{\beta}$ is a vector-valued function. For simplicity, it is assumed that ϵ is a small constant, all results can be extended to the case that ϵ is a symmetric and positive definite matrix-valued function with small entries ϵ_{ij}.

The standard Galerkin method for (1) is to seek $u \in S_0^h$ such that

$$(\epsilon \bigtriangledown u^h, \bigtriangledown v) + (div(\boldsymbol{\beta} u^h) + \alpha u^h, v) = (f, v), \quad \forall v \in S_0^h, \tag{2}$$

where $S_0^h \in H_0^1(\Omega)$ is a finite element space on Ω with zero Dirichlet boundary condition.

To describe the domain decomposition algorithms, we first divide the domain Ω into some nonoverlapping subdomains Ω_i satisfying $\bar{\Omega} = \bigcup_i \bar{\Omega}_i, \quad \Omega_i \bigcap \Omega_j = \emptyset, \quad i \neq j$. Let $S^h(\Omega_i) \subset H^1(\Omega_i)$ be the finite element space on Ω_i, we define

$$V_i = \{v \in S^h(\Omega_i);\ v = 0 \ on \ \ \partial\Omega_i \bigcap \partial\Omega\},$$

$$\hat{S}_0^h = \sum_i V_i = \{v \in S^h(\Omega_i),\ \forall i,\ \ v = 0 \ on \ \partial\Omega\}.$$

Notice that functions from $\hat{S}_0^h$ can have jumps along the interfaces. Bilinear form $A_i(\cdot,\cdot)$ is defined as:

$$A_i(w, v) = (\epsilon \bigtriangledown w, \bigtriangledown v)_{\Omega_i} + (div(\boldsymbol{\beta} w) + \alpha w, v)_{\Omega_i} - \int_{\partial\Omega_i^-} w_+ v_+ \boldsymbol{n}\boldsymbol{\beta} ds,$$

where $\boldsymbol{n}$ is the unit outer normal vector on $\partial\Omega_i$ and

$$w_\pm(x) = \lim_{s\to 0^\pm} w(x + s\boldsymbol{\beta}), \qquad (w, v)_{\Omega_i} = \int_{\Omega_i} wv dx,$$

$$\partial\Omega_i^- = \{x \in \partial\Omega_i,\ \ \boldsymbol{\beta}(x) \cdot \boldsymbol{n}(x) \leq 0\}.$$

Our hybrid domain decomposition finite element solution is to find $\hat{u}^h = \sum \hat{u}_i^h$ such that $\hat{u}_i^h = 0$ in $\Omega \setminus \Omega_i$, and in Ω_i, $\hat{u}_i^h \in V_i$ satisfies

$$A_i(\hat{u}_i^h, v) = (f, v)_{\Omega_i} - \int_{\partial\Omega_i^-} (\hat{u}^h)_- v_+ \boldsymbol{n}\boldsymbol{\beta} ds, \quad \forall v \in V_i, \tag{3}$$

where $(\hat{u}^h)_-$ is the boundary value of the solution of the adjacent subdomains in the upwind direction.

In order to solve the subdomain problem (3) to get $\hat{u}_i^h$, the inflow boundary condition $\hat{u}^h|_{\partial\Omega_i^-}$ must be known. Therefore, we need to assume that the flow is simple so that the domain Ω can be divided into subdomains and when the subdomain problems are solved one after another in the flow direction, the inflow boundary condition is always known from the neighbouring subdomains. If the flow does not have closed streamlines, this kind of division is always possible. By suitably organising the subdomains, the computation of the subdomains in the cross-wind direction can be done in parallel.

In the domain decomposition scheme (3), an artificial boundary condition on the outflow boundary is introduced, i.e. we are in fact using

$$\frac{\partial u}{\partial n} = 0, \ \text{on} \ \partial\Omega_i^+,\ \forall i. \tag{4}$$

An error will be produced by this artificial boundary condition. It can be proved, see [ETY96], that when ϵ is small, the effect from the artificial boundary condition is small. This is also confirmed in our numerical experiments.

Theorem 2.1 *Suppose u is the solution of (1), $\hat{u}^h$ is the hybrid domain decomposition finite element solution of (3), $\alpha + \frac{1}{2}div\boldsymbol{\beta} \geq \gamma > 0$ in Ω, and $|\boldsymbol{n\beta}| \geq \gamma_1 > 0$ on inner boundaries $\partial\Omega_i^- \backslash \partial\Omega$, $\forall i$, then*

$$\|u - \hat{u}^h\|_0 \leq C(\|u - u^I\|_1 + \epsilon \sum_i \|\frac{\partial u}{\partial n}\|_{0,\partial\Omega_i^- \backslash \partial\Omega}) , \tag{5}$$

where $u^I \in S_0^h$ is the interpolation of the solution u, C is a positive constant independent of h, ϵ and u.

Remark 2.2 *Compare (5) with the standard error estimate, one sees that the error resulted from the artificial boundary condition is only*

$$O(\epsilon)(\sum_i \|\frac{\partial u}{\partial n}\|_{0,\partial\Omega_i^- \backslash \partial\Omega}). \tag{6}$$

For convection-dominated problems, boundary layers and transient layers can appear inside the domain Ω. In getting the subdomains, we shall avoid the situation that the subdomain boundaries are parallel to the streamlines in the singular layers. Outside the singular layers, there is no problem. Due to the reason that the boundary layers are always narrow, i.e. of width less or equal $O(\epsilon)$, we can construct the subdomains in such a way that the part of $\partial\Omega_i^-$ contained in the singular layers is only of size $O(\epsilon)$. Then, in the worst case, the summation of the error from all the subdomains is

$$\epsilon \sum_i \|\frac{\partial u}{\partial n}\|_{\partial\Omega_i^- \backslash \partial\Omega} = O(\epsilon^{\frac{1}{2}}). \tag{7}$$

If linear finite elements are used for the approximation and the mesh size is h in the part of the domain where the solution is smooth, then the error caused by the artificial boundary condition is negligible when $\epsilon \leq O(h^2)$.

Remark 2.3 *If $\alpha = 0$, $\beta =$ constant, then condition $\alpha + \frac{1}{2}div\boldsymbol{\beta} \geq \gamma > 0$ is not satisfied. Theorem 2.1 is still correct if we just replace $\|u - \hat{u}^h\|_0$ by $\|u - \hat{u}^h\|_A$. Here*

$$\|v\|_A^2 = \sum_i (\epsilon \nabla v, \nabla v)_{\Omega_i} + \frac{1}{2} \sum_i \int_{\partial\Omega_i^-} [v]^2 |\boldsymbol{n\beta}| ds ,$$

and $[v]$ denotes the jump of v on the inflow boundaries. So, we cannot control the errors in the L^2 norm, instead we can only control the errors on the subdomain boundaries, see [ETY96].

Remark 2.4 *The streamline diffusion finite element method (SDFEM) is stable and shall be used to compute the subdomain solutions preferably. Corresponding error estimate can also be obtained for SDFEM, see [ETY96].*

3 Some Discussions and Extensions

An Iterative Domain Decomposition Method

When the flow is complicated and there are closed streamlines, it could be difficult to construct the subdomains in such a way that the subdomain solutions can be computed in the flow direction and inflow boundary condition is always available when we come to compute the solution of a new subdomain. In this case, we only need to construct the subdomains to guarantee $\boldsymbol{n\beta} \geq \gamma_1 > 0$ on $\partial\Omega_i^-$, $\forall i$. Now, the subdomain solutions are all coupled to each other. An iterative scheme is needed. During the iteration, the inflow boundary condition is taken from the previous iterative step, and the algorithm can be written as:

Algorithm 1
Step 1. Choose initial value $\hat{u}_h^0$;
Step 2. For $n \geq 1$, in every subdomain Ω_i, find $\hat{u}_h^{n+1}|_{\Omega_i} = \hat{u}_i^{n+1} \in V_i$ such that

$$A_i(\hat{u}_i^{n+1}, v) = (f, v) - \int_{\partial\Omega_i^-} (\hat{u}_h^n)_- v_+ \boldsymbol{n\beta} ds, \quad \forall v \in V_i \ ; \tag{8}$$

Step 3. Go to the next iteration.

For the above scheme, it can be proved, see [ETY96], that when $\hat{u}^h$, $\hat{u}_h^n$ are the solutions of (3) and (8), $\alpha + \frac{1}{2} div\boldsymbol{\beta} \geq 0$, then the iterative scheme (8) is convergent, i.e. $\|\hat{u}^h - \hat{u}_h^n\|_0 \to 0$ as $n \to \infty$, and when $\alpha + \frac{1}{2} div\boldsymbol{\beta} \geq \gamma > 0$, the spectral radius of the iteration operator T_0 satisfies $\rho(T_0) \leq \left(\dfrac{1}{1 + C\epsilon + C\gamma h}\right)^{\frac{1}{2}}$.

Remark 3.1 *When ϵ is not small, different kinds of boundary condition on the outflow boundary should be used to improve the accuracy. For example, Lagrange multiplier can be used on the inner boundaries, see [WY97] for the details.*

Time-dependent Problems

Consider the time dependent convection-diffusion problem:

$$\begin{cases} u_t - div(\epsilon \bigtriangledown u) + div(\boldsymbol{\beta} u) = f, \text{ in } \Omega \times [0, T], \\ u(x, t) = 0, \text{ on } \partial\Omega \times [0, T], \quad u(x, 0) = u_0(x), \text{ in } \Omega. \end{cases}$$

We can use the backward difference scheme for t. In every time step, we just need to solve

$$-div(\epsilon \bigtriangledown \bar{u}^{k+1}) + div(\boldsymbol{\beta} \bar{u}^{k+1}) + \frac{\bar{u}^{k+1}}{\Delta t} = f + \frac{\bar{u}^k}{\Delta t}, \text{ in } \Omega, \tag{9}$$

with $\bar{u}^{k+1} = 0$ on $\partial\Omega$. Problem (9) is the same kind of problem as (1) with $\alpha = \frac{1}{\Delta t}$. So, the domain decomposition schemes can be used to solve (9). Similar to theorem 2.1, it can be proved that

$$\|\bar{u}^k - \hat{u}_h^k\|_0 \leq C(\epsilon\sqrt{\Delta t} \sum_i (\|\frac{\partial u}{\partial n}\|_{0,\partial\Omega_i^- \backslash \partial\Omega}) + \Delta t \ \|u - u^I\|_1 + \|u - u^I\|_0), \ \ \forall k,$$

where $\hat{u}_h^k$ is the domain decomposition solution of (9) by the non-iterative scheme (3). If the iterative domain decomposition is used to solve (9), because $\alpha = \frac{1}{\triangle t}$ is large, $\alpha + \frac{1}{2} div\boldsymbol{\beta} \geq 0$, so the iteration is convergent, and the spectral radius of the iteration operator is $\rho(T_0) \leq \left(\frac{1}{1 + C\epsilon + Ch(\triangle t)^{-1}}\right)^{\frac{1}{2}}$. Hence, when $\triangle t = O(h)$, $\rho(T_0) \leq C < 1$, i.e. the error reduction of the iteration is uniform. Especially, when $\triangle t = O(h^2)$, $\rho(T_0) \leq C\sqrt{h}$, therefore only a few iteration steps are required at every time level, see [ETY96] for the analyses.

4 Numerical Experiments

As a test example, we solve the model problem

$$-\epsilon \triangle u + divu + 2u = f, \text{ in } \Omega, \tag{10}$$

with $\Omega = [0,1] \times [0,1]$ and $u = 0$ on $\partial\Omega$. We choose

$$f = C_1(e^{a(1-x)} + e^{a(1-y)}) + C_2(e^{b(1-x)} + e^{b(1-y)}) + 2$$

with

$$C_1 = \frac{1 - e^b}{e^b - e^a}, \quad C_2 = \frac{e^a - 1}{e^b - e^a}, \quad a = \frac{-1 + \sqrt{1 + 4\epsilon}}{2\epsilon}, \quad b = \frac{-1 - \sqrt{1 + 4\epsilon}}{2\epsilon}.$$

Then the analytical solution is

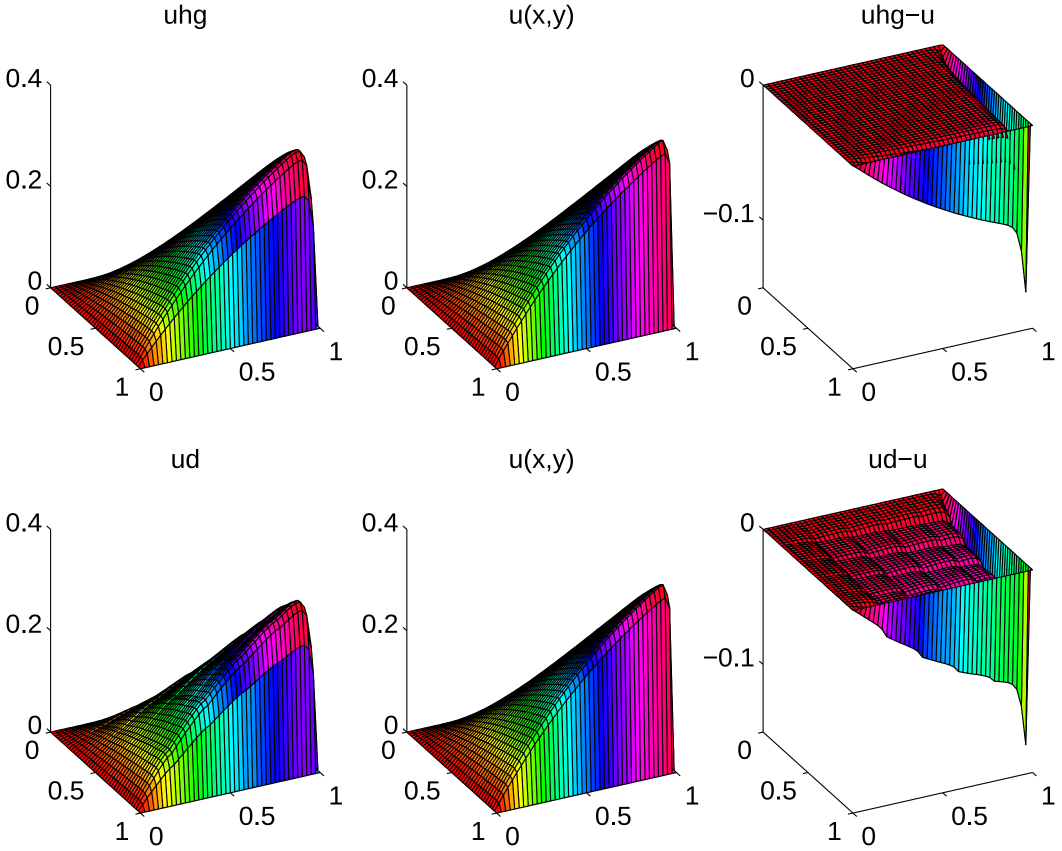
$$u = (C_1 e^{a(1-x)} + C_2 e^{b(1-x)} + 1)(C_1 e^{a(1-y)} + C_2 e^{b(1-y)} + 1).$$

In the computations, the domain Ω is divided into 5×5 subdomains. Piecewise linear finite element functions on uniform triangular meshes is used. In each subdomain, a first order upwind approximation is used for the convection term and the inflow boundary condition is realised exactly which is taken from the subdomains in the upwind direction. Let i=1,2,3,4,5, and j=1,2,3,4,5 be the numbers associated with the subdomains in the x- and y-directions. We solve the subdomain problems by first sweeping over i=1,2,3,4,5 and then sweeping over j=1,2,3,4,5. By solving the subdomain problems in this order, the inflow boundary condition is always available when we come to compute a subdomain solution.

In table 1, some numerical results for different ϵ and different mesh sizes h are shown, where $\|e_g\|_0$ and $\|e_d\|_0$ represent the error of the global finite element solution and the error of the domain decomposition solution for problem (10) in L^2-norm, respectively.

Figure 1 shows the computed solutions and their errors for $\epsilon = 0.01$ and $h = 0.025$, where u, uhg and ud represent the exact solution, the global finite element solution and the domain decomposition solution of (10), respectively. From table 1 and figure 1, one observes that when ϵ is small, the error of the domain decomposition solution is of the same order as the global finite element solution (see Table 1 for ϵ =0.01, 0.001, 0.00001). From figure 1, one finds that the large errors both for the global FEM solution and the domain decomposition solution are concentrated in the neighbourhood of

Figure 1 The global FEM solution and the domain decomposition solution for $\epsilon = 0.01,\ h = 0.025$ and the corresponding errors.



the outflow boundary. Due to the relative large mesh size used near the outflow boundary, the boundary layer is not properly resolved. Here comes the advantage of the proposed domain decomposition methods. Once we know that which subdomain contains the singular layers, we can use finer mesh in this subdomain. By doing so, the error introduced by the artificial boundary condition does not increase, but the singular layers can be efficient resolved by using the known boundary conditions from the neighbouring subdomains and a sufficient fine mesh in this subdomain. Different examples have been tested by the proposed algorithms. The numerical results always show that when ϵ is small, the domain decomposition solution and the global finite element solution have errors of the same order and the large errors are in the singular layers. To do grid refinement for the global problem is not easy, but it is very easy to use fine meshes for the subdomains that contains the singular layers.

Table 1. L^2-error of the global solution and the domain decomposition solution.

	$\epsilon = 0.1$		$\epsilon = 0.01$		$\epsilon = 0.001$		$\epsilon = 0.00001$	
	$\Vert e_g \Vert_0$	$\Vert e_d \Vert_0$	$\Vert e_g \Vert_0$	$\Vert e_d \Vert_0$	$\Vert e_g \Vert_0$	$\Vert e_d \Vert_0$	$\Vert e_g \Vert_0$	$\Vert e_d \Vert_0$
h=0.1	0.0155	0.0354	0.0126	0.0200	0.0066	0.0095	0.0061	0.0084
h=0.05	0.0085	0.0323	0.0131	0.0186	0.0038	0.0051	0.0029	0.0035
h=0.025	0.0045	0.0297	0.0122	0.0165	0.0029	0.0037	0.0014	0.0016
h=0.0125	0.0023	0.0280	0.0082	0.0122	0.0032	0.0036	0.0007	0.0007

5 Conclusion

Both theoretical analysis and numerical tests reveal that the proposed algorithms are suitable for problems with small ϵ. When the diffusion parameter is small, the singular layers are very narrow. In order to resolve the singular layers, the ratio between the mesh size in the singular layers and the mesh size in the part of the domain where the solution is smooth shall be very large. In this case, the error introduced by the domain decomposition algorithms are negligible in comparison with the errors in the singular layers. However, the domain decomposition algorithms allow easy and efficient grid refinement in the subdomains that contain the singular layers.

Acknowledgement

This work is partially supported by the University of Bergen and by the Research Council of Norway.

REFERENCES

[ETY96] Espedal M., Tai X.-C., and Yan N. (1996) A hybrid domain decomposition method for advection-diffusion problems. Technical Report 102, Department of Mathematics, University of Bergen.

[KL95] Kapurkin A. and Lube G. (1995) A domain decomposition for singular perturbed elliptic problems. In Hackbusch W. and Wittum G. (eds) *Notes on Numerical Fluid Mechanics*, volume 49, pages 151–162. Vieweg Verlag, Stuttgart.

[RT97] Rognes Ø. and Tai X.-C. (1997) A space decomposition method for nonsymmetric problems. *To appear* .

[RZ94] Rannacher R. and Zhou G. H. (1994) Analysis of a domain–splitting method for nonstationary convection-diffusion problems. *East-West J. Numer. Math.* 2: 151–174.

[TJDE97] Tai X.-C., Johansen T., Dahle H. K., and Espedal M. (1997) A characteristic domain splitting method. In *The proceeding of the 8th international domain decomposition conference (to appear).*

[WY97] Wang J. P. and Yan N. N. (1997) A parallel domain decomposition procedure for advection diffusion problems. In *The proceeding of the 8th international domain decomposition conference (to appear).*

28

Geometric Convergence of Overlapping Schwarz Methods for Obstacle Problems

Jinping Zeng

1 Obstacle Problem and its Discretization

Schwarz methods have been paid great attention in recent years. In our paper, we consider Schwarz methods for the obstacle problem of finding a u such that

$$\begin{cases} -\Delta u(x) \geq f(x), & \\ u(x) \geq 0, & x \in \Omega, \\ u(x)(-\Delta u(x) - f(x)) = 0, & \\ u(x) = g(x), & x \in \partial\Omega, \end{cases} \tag{1.1}$$

where Ω is a bounded polyhedron convex domain in R^2 or R^3 with boundary $\partial\Omega$. f and g are given functions.

We discrete problem (1.1) as finite-dimensional linear complementarity problem by using a conforming finite element method (e.g. Lagrange linear elements):

$$\begin{cases} LU(x) \geq f(x), \quad U(x) \geq 0 \quad U^T(x)(LU(x) - f(x)) = 0. & x \in \Omega_h, \\ U(x) = g(x), & x \in \partial\Omega_h, \end{cases}$$

where Ω_h is grid set of the triangulation of Ω. Let grid function U be the restriction of $U(x)$ on Ω_h. Then we have that

$$AU \geq F, \quad U \geq 0, \quad U^T(AU - F) = 0. \tag{1.2}$$

Where A is a symmetric positive definite M-matrix if each angle of the mesh is an acute angle.

In [KNT94, KNT95] the numerical solution of linear complementarity problems with monotone operators by relaxation and Schwarz-type overlapping domain decomposition methods are considered. Monotone convergence was obtained for a special initial choice. Similar convergence was also discussed in [Sca90, Zho96]. Until now, however, we have not seen any discussion on the convergence rate except in

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

[Tar96, ZZ]. In [ZZ], we analyzed convergence rate of the algorithm for solving obstacle problem if we confine the initial value to as special set, the same as [KNT94] or [KNT95]. Here, without any limitation of initial value, we prove that the iterate sequence generated by the algorithm we proposed converges to the solution of (1.2).

2 Schwarz Algorithm

Following the substructuring idea given by Dryja and Widlund (c.f. [Dry89, DW90, LSL92]), we can construct Schwarz algorithm for solving (1.2): We use the two-level triangulation of Ω given in [LSL92]. In this way we get the H-level triangulation consisting of Ω_i with diameters $H_i (i = 1, \dots, m)$ and the overlapping open subdomains $\Omega'_i \supset \Omega_i (i = 1, \dots, m)$. We assume that there is a positive constant c such that

$$\text{dist } (\partial\Omega'_i \backslash \partial\Omega, \partial\Omega_i \backslash \partial\Omega) \geq cH_i, \quad i = 1, \dots, m. \tag{2.1}$$

Let $\Omega_{ih} (i = 1, \dots, m)$ be the sets of the nodes that belong to open subdomains Ω'_i respectively, N_i be the subset of index set $\{1, 2, \dots, N\}$: $N_i = \{j \in \{1, 2, \dots, N\} : x_j \in \Omega_{ih}\}$. $A_{I,J}$ denotes the submatrix of A with elements a_{ij} $(i \in I, j \in J)$, U_I denotes the subvector of U with elements U_i $(i \in I)$. Then additive Schwarz algorithm can be stated as follows:

Additive Schwarz Algorithm

Step 1. $n := 0$.

Step 2. For $i = 1, 2, \dots, m$,

$$\begin{cases} A_{N_i,N_i} U_{N_i}^{n+1,i} \geq F^{n,i}, \\ U_{N_i}^{n+1,i} \geq 0, \\ (U_{N_i}^{n+1,i})^T (A_{N_i,N_i} U_{N_i}^{n+1,i} - F^{n,i}) = 0. \end{cases} \tag{2.2}$$

$$U_{N \backslash N_i}^{n+1,i} = U_{N \backslash N_i}^n, \tag{2.3}$$

where

$$F^{n,i} = F_{N_i} - A_{N_i, N \backslash N_i} U_{N \backslash N_i}^n. \tag{2.4}$$

Step 3. Choose ω_i satisfying

$$0 < \omega_i < 1, \quad i = 1, 2, \dots, m, \quad \sum_{i=1}^m \omega_i = 1.$$

Let

$$U^{n+1} = \sum_{i=1}^m \omega_i U^{n+1,i}. \tag{2.5}$$

Step 4. $n := n + 1$, go to step 2.

3 Geometric Convergence

Lemma 3.1(iteration error estimate) *Let $\epsilon^{n+1,i} = U^{n+1,i} - U$, $\epsilon^n = U^n - U$ (where U is the solution of (1.2)). Then for all $n = 0, 1, \dots$ and $i = 1, \dots m$, we have that*

$$(A|\epsilon^{n+1,i}|)_{N_i} \leq 0. \tag{3.1}$$

Proof. If $(U_{N_i})_k = 0$ and $(U_{N_i}^{n+1,i})_k > 0$, then

$$(A_{N_i,N_i}U_{N_i}^{n+1,i} - F^{n,i})_k = 0. \tag{3.2}$$

Let

$$F^{*,i} = F_{N_i} - A_{N_i,N\backslash N_i}U_{N\backslash N_i}.$$

We have

$$(A_{N_i,N_i}U_{N_i} - F^{*,i})_k \geq 0. \tag{3.3}$$

If we subtract (3.3) from (3.2), we get that

$$(A_{N_i,N_i}\epsilon_{N_i}^{n+1,i})_k \leq (-A_{N_i,N\backslash N_i}\epsilon_{N\backslash N_i}^{n})_k.$$

Since $A_{kj} \leq 0$ for $k \neq j$ and (2.3), we have

$$(A_{N_i,N_i}|\epsilon_{N_i}^{n+1,i}|)_k \leq (-A_{N,N\backslash N_i}|\epsilon_{N\backslash N_i}^{n+1,i}|)_k.$$

i.e.

$$((A|\epsilon^{n+1,i}|)_{N_i})_k \leq 0.$$

If $(U_{N_i})_k > 0$ and $(U_{N_i}^{n+1,i})_k > 0$, then

$$(A_{N_i,N_i}U_k^{n+1,i} - F^{n,i})_k = 0,$$

and

$$(A_{N_i,N_i}U_{N_i} - F^{*,i})_k = 0.$$

Hence

$$(A_{N_i,N_i}\epsilon_{N_i}^{n+1,i})_k = (-A_{N_i,N\backslash N_i}\epsilon_{N\backslash N_i}^{n})_k.$$

Therefore,

$$(A_{N_i,N_i}|\epsilon_{N_i}^{n+1,i}|)_k \leq (-A_{N,N\backslash N_i}|\epsilon_{N\backslash N_i}^{n}|)_k = (-A_{N_i,N\backslash N_i}|\epsilon_{N\backslash N_i}^{n+1,i}|)_k.$$

If $(U_{N_i})_k > 0$ and $(U_{N_i}^{n+1,i})_k = 0$, then

$$(A_{N_i,N_i}U_k^{n+1,i} - F^{n,i})_k \geq 0,$$

and

$$(A_{N_i,N_i}U_{N_i} - F^{*,i})_k = 0.$$

Then

$$(A_{N_i,N_i}\epsilon_{N_i}^{n+1,i})_k \geq (-A_{N_i,N\backslash N_i}\epsilon_{N\backslash N_i}^{n})_k.$$

Since $(\epsilon_{N_i}^{n+1,i})_k < 0$, we also have

$$(A_{N_i,N_i}|\epsilon_{N_i}^{n+1,i}|)_k \leq (-A_{N,N\backslash N_i}|\epsilon_{N\backslash N_i}^{n}|)_k = (-A_{N_i,N\backslash N_i}|\epsilon_{N\backslash N_i}^{n+1,i}|)_k.$$

That is (3.1) holds. Thus we complete the proof.

In order to prove the geometric convergence, we also need to establish the discrete strong maximum principle as follows:

Lemma 3.2 *For finite element discretization, if each angle of the mesh is an acute angle and the boundary of every subdomain has a common part with the boundary of the whole domain. Then, for any $V \geq 0$ satisfying $(AV)_{N_i} \leq 0$, there exists a constant $k_i \in (0,1)$ such that*

$$\max_{j \in N_i} V_j \leq k_i \max_{j \notin N_i} V_j.$$

Remark Under the conditions of lemma 3.2, matrix A has the following properties: (i) $a_{ii} > 0$, $a_{ij} \leq 0 (j \neq i)$; (ii) A is irreducible and weak diagonal dominant; (iii) $A_{N_i N_i}$ are irreducible, $i = 1, \ldots, m$; (iv) for every $i = 1, \ldots, m$, there exists $l_i \in N_i$ such that $\sum_{j=1}^{N} a_{l_i j} > 0$. Therefore, lemma 3.1 can be proved by reduction to absurdity.

By lemma 3.1 and lemma 3.2 we get the following geometric convergence result:

Theorem 3.1 *Under the conditions of lemma 3.2, we have that*

$$\|\epsilon^{n+1}\|_\infty \leq \max_{1 \leq i \leq m} (\omega_i k_i + \sum_{j \neq i} \omega_j) \|\epsilon^n\|_\infty = k \|\epsilon^n\|_\infty, \tag{3.4}$$

where $k = \max_{1 \leq i \leq m} (\omega_i k_i + \sum_{j \neq i} \omega_j) \in (0,1)$.

Proof $\forall k \in \{1, 2, \ldots, N\}$ and $i \in \{1, 2, \ldots, m\}$. If $k \notin N_i$, then $|\epsilon^{n+1,i}|_k = |\epsilon^n|_k \leq \|\epsilon^n\|_\infty$; If $k \in N_i$, since $(A|\epsilon^{n+1,i}|)_{N_i} \leq 0$. By use of lemma 3.1 and lemma 3.2, we have that

$$|\epsilon^{n+1,i}|_k \leq k_i \max_{j \notin N_i} |\epsilon^{n+1,i}|_j = k_i \max_{j \notin N_i} |\epsilon^n|_j \leq k_i \|\epsilon^n\|_\infty.$$

Since $\{1, 2, \ldots, N\} = \bigcup_{1}^{m} N_i$, we get (3.4).

4 Concluding Remarks

The goal of the paper is to give the convergence proof of additive Schwarz algorithm for the algebraic obstacle problems by establishing discrete maximum principle. The convergence theory can be found useful for many resons. First, Our results are suitable for obstacle problems with nonsymmetric operators. As we know, many discussions about Schwarz methods for obstacle problems require the problems have self-adjoint and positive definite operators (c.f. [Lio88, Lio89, Sca90, Xu92, Zho96]). Second, we can get geometric convergent rate by establishing corresponding discrete maximum principle. Especially, we can estimating h-independent convergence of the additive Schwarz algorithm by estimating k_i in lemma 3.2 (see [ZZ] for details). Finally, we notice that the convergence theory of this paper can be extended to multiplicative Schwarz algorithm or other obstacle problems as well. For example, using a similar analysis, we could prove geometric convergence of additive or multiplicative algorithms applied to solving bilateral discrete obstacle problems. These extensions will be studies in the forthcoming publications.

Acknowledgement

The author would like to thank Prof. Shuzi Zhou for his valuable comments on the paper and the Organizing Committee of DD9 and NNSF of China for their financial support.

REFERENCES

[Dry89] Dryja M. (1989) An additive Schwarz algorithm for two-and three-dimensional finite element elliptic problems. In Chan T., Glowinski R., Périaux J., and Widlund O. B. (eds) *Domain Decomposition Methods*, pages 168–172. SIAM, PA, Philadelphia.

[DW90] Dryja M. and Widlund O. B. (1990) Towards a unified theory of domain decomposition algorithms for elliptic problems. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Domain Decomposition Method for PDE*, pages 340–349. SIAM, Philadelphia.

[KNT94] Kuznetsov Y., Neittaanmäki P., and Tarvainen P. (1994) Block relaxation methods for algebraic obstacle problem with m-matrices. *East-West Journal of Numerical Mathematics* (2): 75–89.

[KNT95] Kuznetsov Y., Neittaanmäki P., and Tarvainen P. (1995) Schwarz methods for obstacle problems with convection-diffusion operators. In Keyes D. E. and Xu J. C. (eds) *Domain Decomposition Methods in Scientifical and Engineering Computing*, pages 251–256. AMS.

[Lio88] Lions P. L. (1988) On the Schwarz alternating method I. In Glowinski R., Golub G. H., Meurant G. A., and Périaux J. (eds) *Domain Decomposition Method*, pages 1–40. SIAM.

[Lio89] Lions P. L. (1989) On the Schwarz alternating method II. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Domain Decomposition Method*, pages 47–70. SIAM.

[LSL92] Lü T., Shih T. M., and Liem C. B. (1992) *Domain Decomposition Methods—New Numerical Technique for solving PDE (in Chinese).* Academic Press, Beijing.

[Sca90] Scarpini F. (1990) The alternative Schwarz method applied to some biharmonic variational inequalities. *Calcolo* 27: 57–72.

[Tar96] Tarvainen P. (1996) On the convergence of block relaxation methods for algebraic obstacle problems with m-matrices. *East-West Journal of Numerical Mathematics* (4): 69–82.

[Xu92] Xu J. C. (1992) Iterative methods by space decompositiion and subspace correction. *SIAM Review* 34: 581–613.

[Zho96] Zhou S. Z. (1996) An additive Schwarz algorithm for variational inequality. In Glowinski R., Périaux J., Shi Z., and Widlund O. B. (eds) *Domain Decomposition Methods in Science and Engineering.*

[ZZ] Zeng J. P. and Zhou S. Z.On monotone and geometric convergence of Schwarz methods for two-side obstacle problems. *SIAM Journal on Numerical Analysis (to appear) .*

29

A Multigrid Iterated Penalty Method for Mixed Elements

Ridgway Scott and Shangyou Zhang

1 Introduction

We consider the following stationary Stokes problem: Find functions $\mathbf{u}$ (the fluid velocity) and p (the pressure) on a bounded polygonal/polyhedral domain $\Omega \subset \mathbf{R}^d$ $(d = 2, 3)$, such that

$$\begin{aligned} -\Delta\mathbf{u} + \mathrm{grad} p &= \mathbf{f} && \text{in } \Omega, \\ \mathrm{div}\mathbf{u} &= 0 && \text{in } \Omega, \\ \mathbf{u} &= 0 && \text{on } \partial\Omega, \end{aligned} \tag{1}$$

where $\mathbf{f}$ is the body force. In the variation form of velocity-pressure formulation of the Stokes equations, the velocity and pressure are in the Sobolev spaces $H_0^1(\Omega)^d$ and $L_0^2(\Omega)$, respectively. The mixed element approximation spaces can be chosen to be the corresponding subspaces. A natural pairing would be continuous piecewise-polynomials of degree $(k+1)$ and discontinuous piecewise-polynomials of degree k for the velocity and pressure, respectively. Such mixed element solutions satisfy the incompressibility condition, i.e. pointwise divergence free. Scott and Vogelius [SV85] showed that the Babuška-Brezzi inequality holds for such $\mathcal{P}_{k+1}$-P_k triangular mixed-elements in 2D if the polynomial degree k is 3 or higher and if the meshes are singular-vertex free. This result is partially extended to 3D in [Zha94]. It is shown that, when defined on tetrahedral meshes of a macro-element type, the above $\mathcal{P}_{k+1}$-P_k elements are stable if the polynomial degree for velocity is 3 or higher.

The mixed elements approximation to (1) in weak formulation is: Find $[\mathbf{u}_j, p_j] \in V_j$, such that

$$a(\mathbf{u}_j, \mathbf{v}) + b(\mathbf{v}, p_j) + b(\mathbf{u}_j, q) = (\mathbf{f}, \mathbf{v}) \quad \forall [\mathbf{v}, q] \in V_j, \tag{2}$$

where $a(\mathbf{u}, \mathbf{v}) := (\nabla\mathbf{u}, \nabla\mathbf{v})$ and $b(\mathbf{v}, p) := -(\mathrm{div}\mathbf{v}, p)$. Here $\{V_j\}$ are multilevel $\mathcal{P}_{k+1}$-P_k mixed finite element spaces:

$$V_j = (\mathcal{P}^0_{k+1,\mathcal{T}_j})^d \times P^0_{k,\mathcal{T}_j} \subset (H_0^1(\Omega))^d \times L_0^2(\Omega),$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

defined on nested or nonnested triangular/tetrahedral grids $\{\mathcal{T}_j\}$ covering Ω, where $\mathcal{P}^0_{k,\mathcal{T}_j} = \mathcal{P}_{k,\mathcal{T}_j} \cap H^1_0(\Omega)$, $P^0_{k,\mathcal{T}_j} = P_{k,\mathcal{T}_j} \cap L^2_0(\Omega)$, $\mathcal{P}_{k,\mathcal{T}_j}$ and $P_{k,\mathcal{T}_j}$ are the piecewise continuous and discontinuous polynomials of degree k on the mesh $\mathcal{T}_j$, respectively.

We note that for $\mathcal{P}_{k+1}$-P_k elements, the divergence of the discrete velocity space is precisely the discrete pressure space, i.e., $\{\text{div}\mathbf{u} \mid \mathbf{u} \in (\mathcal{P}^0_{k+1,\mathcal{T}_j})^d\} = P^0_{k,\mathcal{T}_j}$. Therefore, we can avoid introducing the discrete space $P^0_{k,\mathcal{T}_j}$ in computation completely in the following iterated penalty method (3), which is a special form of the augmented Lagrangian method [FG83] . Let $r \geq 1$. The iterated penalty method [BS94] defines $\mathbf{u}^n \in (\mathcal{P}^0_{k+1,\mathcal{T}_j})^d$ by

$$\begin{aligned} a(\mathbf{u}^n, \mathbf{v}) + r(\text{div}\mathbf{u}^n, \text{div}\mathbf{v}) &= (\mathbf{f}, \mathbf{v}) + (\text{div}\mathbf{v}, \text{div}\mathbf{w}^n) \qquad \forall \mathbf{v} \in (\mathcal{P}^0_{k+1,\mathcal{T}_j})^d, \\ \mathbf{w}^{n+1} &= \mathbf{w}^n + r\mathbf{u}^n \end{aligned} \tag{3}$$

sequentially given $\mathbf{w}^0 \in (\mathcal{P}^0_{k+1,\mathcal{T}_j})^d$ (which is usually $\mathbf{0}$). $p^n = \text{div}\mathbf{w}^n \in P^0_{k,\mathcal{T}_j}$. The key point of the iterated penalty method is that the system of equations represented by the first equation in (3) for $\mathbf{u}^n$ will be symmetric and positive definite. Each time when we solve the linear system (3), we apply the multigrid method to get an inner iteration. Another essential point here is that we only do one or two multigrid iterations for (3), which needs not to be solved accurately as the solution $\mathbf{u}^n$ is not final. We show that the overall convergence rate of the combined iterated penalty – multigrid method is independent of the size of the discrete system, and that the overall computation cost for solving the discrete Stokes equations up to the order of approximation is proportional to the system size if the penalty parameter r is chosen not too big. Therefore the method is optimal in the order of computation. The analysis is confirmed by our numerical computation.

2 Analysis

We first analyze the the iterated penalty method. We will define a multigrid iterated penalty method. Then we will apply the general theory developed in [BP87], [BPX91] and [SZ92] to obtain a convergence analysis of the multigrid iterated penalty method.

Theorem 1. (Convergence of the iterated penalty method) *For $\mathbf{u}^n$ defined in (3), the following error reduction relation holds:*

$$|||\mathbf{u}_j - \mathbf{u}^n|||_{1,j} \leq \frac{C}{r} |||\mathbf{u}_j - \mathbf{u}^{n-1}|||_{1,j}$$

where C is independent of j.

PROOF. Let the errors be denoted as $\mathbf{e}^n = \mathbf{u}^n - \mathbf{u}_j$ and $\epsilon^n = p^n - p_j = \text{div}\mathbf{w}^n - p_j$. We will use a short notation

$$\mathbf{U}_j := (\mathcal{P}^0_{k+1,\mathcal{T}_j})^d.$$

Subtracting (2) from (3), it follows

$$a(\mathbf{e}^n, \mathbf{v}) + r(\text{div}\mathbf{e}^n, \text{div}\mathbf{v}) = b(\mathbf{v}, \epsilon^n) \quad \forall \mathbf{v} \in \mathbf{U}_j, \tag{4}$$

$$\epsilon^{n+1} = \epsilon^n + r\text{div}\mathbf{u}^n = \epsilon^n + r\text{div}\mathbf{e}^n. \tag{5}$$

Hence, combining the above two equations we can get

$$\begin{aligned} a(\mathbf{e}^{n+1}, \mathbf{e}^{n+1}) + r(\mathrm{div}\mathbf{e}^{n+1}, \mathrm{div}\mathbf{e}^{n+1}) &= b(\mathbf{e}^{n+1}, \epsilon^{n+1}) && (6) \\ &= b(\mathbf{e}^{n+1}, \epsilon^{n}) - r(\mathrm{div}\mathbf{e}^{n+1}, \mathrm{div}\mathbf{e}^{n}) \\ &= a(\mathbf{e}^{n+1}, \mathbf{e}^{n}) \le |\mathbf{e}^{n+1}|_{H^1}|\mathbf{e}^{n}|_{H^1}. && (7) \end{aligned}$$

By (4), $a(\mathbf{e}^n, \mathbf{v}) = 0$ for all divergence-free function $\mathbf{v} \in \mathbf{U}_j$. Therefore, for our special choices of $\mathbf{U}_j$, we have

$$(\nabla \times \mathbf{e}^n, \nabla \times \mathbf{e}^n) = 0 \quad \text{and} \quad a(\mathbf{e}^n, \mathbf{e}^n) = (\mathrm{div}\mathbf{e}^n, \mathrm{div}\mathbf{e}^n).$$

The theorem is proven with $C = 1$. ▮

By Theorem 1, we can see that $|||\mathbf{u}_j - \mathbf{u}^n|||_{1,j} = O(r^{-n})$. One may like to choose a very large r to get a fast convergence. However, large n will cause a bad conditioning for the linear system in (3), which will in turn increase the work of the multigrid method when solving the linear system.

In the multigrid method, we have two steps, the fine-level smoothing and the coarse-level correction. The fine-level smoothing is usually the Richardson iteration, where we can introduce an L^2-equivalent discrete inner-product, $(\cdot,\cdot)_j$, on $\mathbf{U}_j$ defined by the diagonal entries of the L^2 inner-product for the nodal basis. We note the equivalence constant depends on the polynomial degree, $k+1$, but is independent of the level number j. We start by defining a family of symmetric positive definite operators, $A_j : \mathbf{U}_j \to \mathbf{U}_j$,

$$(A_j\mathbf{u}, \mathbf{v})_j = a_r(\mathbf{u}, \mathbf{v}) := a(\mathbf{u}, \mathbf{v}) + r(\mathrm{div}\mathbf{u}, \mathrm{div}\mathbf{v}) \quad \forall \mathbf{u}, \mathbf{v} \in \mathbf{U}_j$$

Let $\rho(A_j)$ be the spectral radius of A_j. Then it is well known by the inverse inequality that

$$\rho(A_j) = Ch_j^{-2}r, \tag{8}$$

where the constant C depends on k but not on j. We assume the following regularity for the solution of the continuous version of the first equation (3):

$$\|\mathbf{u}_g\|_{H^{1+\alpha}(\Omega)^d} \le C\|\mathbf{g}\|_{H^{-1+\alpha}(\Omega)^d} \qquad \forall \mathbf{g} \in H^{-1+\alpha}(\Omega)^d, \tag{9}$$

where $\alpha > 0$ and $\mathbf{u}_g$ is defined by $a_r(\mathbf{u}_g, \mathbf{v}) = (\mathbf{g}, \mathbf{v}) \quad \forall \mathbf{v} \in H_0^1(\Omega)^d$. In (9) the constant C is independent of the penalty parameter r. By the standard finite element theory, we have the following estimate in the energy norm

$$|||\mathbf{u}_g - \mathrm{P}_j\mathbf{u}_g|||_{1,j} \le Crh^s\|\mathbf{u}_g\|_{H^{1+s}(\Omega)^d}, \quad s = \min\{k+1, \alpha\}, \tag{10}$$

where $\mathrm{P}_j\mathbf{u}_g$ denotes the finite element solution for $\mathbf{u}_g$. Here the triple-bar norms are defined by $|||\mathbf{v}|||_{s,j}^2 := (A_j^s\mathbf{v}, \mathbf{v})_j, \quad 0 \le s \le 2$.

Definition 1 (One level j W-cycle symmetric nested/nonnested multigrid iteration).
1. For $j = 1$, the problem (3) or the residual problem (12) below is solved exactly.
2. For $j > 1$, $\mathbf{w}_{2m+1}$ will be generated from the initial guess, $\mathbf{w}_0$ (which is either 0

or a previous solution on the same level, the s below is great than 1), as follows.
2-a. m presmoothings are performed to generate $\mathbf{w}_m$:

$$(\mathbf{w}_l - \mathbf{w}_{l-1}, \mathbf{v})_j = \lambda_j^{-1}(\mathbf{F}(\mathbf{v}) - a_r(\mathbf{w}_{l-1}, \mathbf{v})) \quad \forall \mathbf{v} \in \mathbf{U}_j, \quad l = 1, 2, \ldots, m, \tag{11}$$

where $\rho(A_j)/\lambda_k \leq \omega$ for some fixed ω satisfying $0 < \omega < 2$ and $\mathbf{F}$ is either the functional defined in the right-hand side of (3) or the $\tilde{\mathbf{F}}$ in (12).
2-b. $\mathbf{w}_m$ is corrected by $\mathrm{I}_j q$ to generate w_{m+1}: Let $\bar{q}$ solve the following coarse–level residual problem,

$$a_r(\bar{\mathbf{q}}, \mathbf{v}) = \mathbf{F}(\mathrm{I}_j \mathbf{v}) - a_r(\mathbf{w}_m, \mathrm{I}_j \mathbf{v}) =: \tilde{\mathbf{F}}(\mathbf{v}) \quad \forall \mathbf{v} \in \mathbf{U}_{j-1}. \tag{12}$$

Let $\mathbf{q} \in \mathbf{U}_{j-1}$ be the approximation of $\bar{\mathbf{q}}$ obtained by applying s ($s = 2$) iterations of the $(j-1)$st level multigrid scheme to (12) starting with initial guess zero. Then $\mathbf{w}_{m+1} = \mathbf{w}_m + \mathrm{I}_j q$. Here, $\mathrm{I}_j : \mathbf{U}_{j-1} \to \mathbf{U}_j$ is the usual Lagrange interpolation operator if $\mathbf{U}_{j-1} \not\subset \mathbf{U}_j$, or just an identity operator if $\mathbf{U}_{j-1} \subset \mathbf{U}_j$.
2-c. m postsmoothings of the form (11) are performed to generate w_{2m+1} from w_{m+1}.
▮

Definition 2 (A multigrid iterated penalty method). Let $\tilde{\mathbf{w}}^0 = 0$. $\tilde{\mathbf{u}}^n$ is defined by doing l jth level W-cycle symmetric nested/nonnested multigrid iteration for the problem defined by the first equation in (3) where $\mathbf{w}^n$ being replaced by $\tilde{\mathbf{w}}^n$, i.e., for the equation

$$a(\tilde{\mathbf{u}}^n, \mathbf{v}) + r(\mathrm{div}\tilde{\mathbf{u}}^n, \mathrm{div}\mathbf{v}) = (\mathbf{f}, \mathbf{v}) + (\mathrm{div}\mathbf{v}, \mathrm{div}\tilde{\mathbf{w}}^n) \quad \forall \mathbf{v} \in \mathbf{U}_j. \tag{13}$$

Here the initial guess for the multigrid iteration is $\tilde{\mathbf{u}}^{n-1}$ and $\tilde{\mathbf{u}}^{-1} = \mathbf{0}$. $\tilde{\mathbf{w}}^n$ is defined by $\tilde{\mathbf{w}}^n = \tilde{\mathbf{w}}^{n-1} + r\mathrm{div}\tilde{\mathbf{u}}^n$. ▮

By the assumptions (8–9) and (10) it is standard to verify the "regularity and approximation" assumption introduced by Bramble *et al.* (see (3.2) in [BP87], also [BPX91] and [SZ92]):

$$a_r(\mathbf{u} - \mathrm{P}_{j-1}, \mathbf{u}) \leq C_\beta^2 \left(\frac{|||A_j \mathbf{u}|||_{0,j}^2}{\lambda_j} \right)^\beta a_r(\mathbf{u}, \mathbf{u})^{1-\beta} \quad \forall \mathbf{u} \in \mathbf{U}_j$$

with $C_\beta^2 = Cr^{1+\alpha/2}$ and $\beta = \alpha/2$ (α is defined in (9)). Therefore, we can get the following theorem by the theory of [BP87].

Theorem 2. *The error reduction factor in the $|||\cdot|||_{1,j}$ norm for one jth level W-cycle symmetric nested/nonnested multigrid iteration is bounded by*

$$\gamma := \left(\frac{Cr^{1/\alpha+1/2}}{Cr^{1/\alpha+1/2} + m} \right)^\alpha < 1,$$

where α is introduced in (9). ▮

Theorem 3. *The error reduction factor in the $|||\cdot|||_{1,j}$ norm for the multigrid iterated penalty method defined in Definition 2 can be estimated by*

$$|||\tilde{\mathbf{u}}^{n+1} - \mathbf{u}_j|||_{1,j} \leq \left(\frac{C}{r} + \gamma^l\right) |||\tilde{\mathbf{u}}^n - \mathbf{u}_j|||_{1,j},$$

where γ is defined in Theorem 2.

Proof. Let the errors be denoted by $\tilde{\mathbf{e}}^n = \tilde{\mathbf{u}}^n - \mathbf{u}_j$. Let $\bar{\mathbf{u}}^n$ be the exact solution of the first equation in (3) with $\mathbf{w}^n$ there being replaced by $\tilde{\mathbf{w}}^n$. Repeat (4 – 6) in the proof for Theorem 1, it follows also that

$$|||\bar{\mathbf{u}}^{n+1} - \mathbf{u}_j|||_{1,j} \leq \frac{C}{r}|||\tilde{\mathbf{u}}^n - \mathbf{u}_j|||_{1,j}. \tag{14}$$

By (14) and Theorems , we get that

$$\begin{aligned} |||\tilde{\mathbf{e}}^{n+1}|||_{1,j} &\leq |||\tilde{\mathbf{u}}^{n+1} - \bar{\mathbf{u}}^{n+1}|||_{1,j} + |||\bar{\mathbf{u}}^{n+1} - \mathbf{u}_j|||_{1,j} \\ &\leq \gamma^s|||\tilde{\mathbf{u}}^n - \bar{\mathbf{u}}^{n+1}|||_{1,j} + |||\bar{\mathbf{u}}^{n+1} - \mathbf{u}_j|||_{1,j} \\ &\leq \gamma^s|||\tilde{\mathbf{e}}^n|||_{1,j} + (1+\gamma^s)|||\bar{\mathbf{u}}^{n+1} - \mathbf{u}_j|||_{1,j}. \\ &\leq \gamma^s|||\tilde{\mathbf{e}}^n|||_{1,j} + \frac{C(1+\gamma^s)}{r}|||\tilde{\mathbf{e}}^n|||_{1,j}. \blacksquare \end{aligned}$$

According to Theorem 3, we have a constant convergence-rate for the multigrid iterated penalty method. By it, with a full multigrid iteration, one gets in a standard way the optimal order of computation for the algorithm (cf. [SZ92]), i.e., the work to solve (2) up to the finite element truncation error is proportional to the number of unknown in the linear system (2).

3 Numerical Test

Table 1 Number of iteration for the iterated penalty method ($l = 1$, $k = 3$, $m = 10$, $r = 2$)

Grid level	2	3	4	5	6
Iteration number	20	9	9	8	8

In our numerical test, we let $\Omega = (0,1) \times (0,1)$. We let $\mathbf{u} = \nabla \times g$ and $p = \Delta g$ be the exact solutions for (1), where $g = 100(x_1 - x_1^2)^2(x_2 - x_2^2)^2$. The first level grid consists of two triangles, and higher level grids are defined by refining the previous level triangles into 4 subtriangles. In our computation, we use the V-cycle, symmetric multigrid iteration with $m = 10$ as the inner iteration. The iteration number of the iterated penalty method is listed in the Table 1 for degree 4 polynomial approximation

Table 2 Number of iteration for the iterated penalty method ($l = 1$, $k = 6$, $m = 10$, $j = 3$)

Penalty parameter r	1/2	1	3	5	10	100
Iteration number	24	10	9	9	12	37

of $\mathbf{u}$ on different grids, where the penalty parameter $r = 2$. The stopping criterion is 2×10^{-7} for $a(\mathbf{e}^n, \mathbf{e}^n)$.

Next we use degree 7 polynomials to approximate the velocity $\mathbf{u}$. In this case, the mixed finite element solutions are exact, the same as $\nabla \times g$. We numerically test the dependence of the iteration number of the iterated penalty method on the penalty parameter r. We can see from Table 2 that for r between 1 and 10, the number of iteration seems to be the least. But with bigger r, we have to increase the smoothing parameter m or use W-cycle multigrid methods to get a more accurate multigrid solution for each penalty problem. This is indicated by Theorem 3. For example, if we let $m = 50$, then the iteration number would become 5 if $r = 100$. But the overall work is about 3 times as much as that for the case $m = 10$ and $r = 5$.

REFERENCES

[BP87] Bramble J. H. and Pasciak J. E. (1987) New convergence estimates for multigrid algorithms. *Math. Comp.* 49: 311–329.

[BPX91] Bramble J. H., Pasciak J. E., and Xu J. (1991) The analysis of multigrid algorithms with nonnested spaces or non-inherited quadratic forms. *Math. Comp.* 56: 1–34.

[BS94] Brenner S. C. and Scott L. R. (1994) *The Mathematical Theory of Finite Element Methods.* Springer-Verlag, New York.

[FG83] Fortin M. and Glowinski R. (1983) *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-value Problems.* North Holland, Amsterdam.

[SV85] Scott L. R. and Vogelius M. (1985) Norm estimates for a maximal right inverse of the divergence operator in spaces of piecewise polynomials. *Math. Modelling Numer. Anal.* 19: 111–143.

[SZ92] Scott L. R. and Zhang S. (1992) Higher dimensional non-nested multigrid methods. *Math. Comp.* 58: 457–466.

[Zha94] Zhang S. (1994) A new family of stable mixed finite elements for the 3D Stokes equations. *Center for the Mathematics of Waves, University of Delaware* Tech. Report 94-18: 457–466.

Part II

Algorithmic techniques

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

30

Hierarchical Basis for the Convection-Diffusion Equation on Unstructured Meshes

Randolph E. Bank and Sabine Gutsch

1 Introduction

The Hierarchical Basis Multigrid Method was originally developed for sequences of refined meshes. Hierarchical basis functions can be constructed in a straightforward fashion on such sequences of nested meshes. The HBMG iteration itself is just a block symmetric Gauß-Seidel iteration applied to the stiffness matrix represented in the hierarchical basis. Because the stiffness matrix is less sparse than when the standard nodal basis functions are used, the iteration is carried out by forming the hierarchical basis stiffness matrix only implicitly. The resulting algorithm is strongly connected to the classical multigrid V-cycle, except that only a subset of the unknowns on each level is smoothed during the relaxation steps [BDY88].

In recent years, we have generalized such bases to completely unstructured meshes, not just those arising from some refinement process. This is done by recognizing the strong connection between the Hierarchical Basis Multigrid Method and an Incomplete LU factorization of the nodal stiffness matrix. This can best be understood in terms of the graph elimination model for Gaussian elimination. This connection is explored in detail in [BX94]. Once this connection is made, it is fairly easy to make a symbolic ILU algorithm for unstructured meshes which mimics the ILU process on a structured mesh leading to the classical hierarchical basis. This symbolic elimination process essentially defines the *supports* of the hierarchical basis functions (or the sparsity structure of the hierarchical basis stiffness matrix).

In the classical case, certain linear combinations of fine grid basis functions are

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

formed, with the combination coefficients derived from the geometry of the mesh. For these special choices, the linear combinations simplify to coarse grid nodal basis functions. In the case of completely unstructured meshes, the coefficients for the linear combinations can also be specified in a natural way using the geometry of the mesh. However, since there is no coarse grid as such, in general no simplification of the basis functions occurs. Different choices of expansion coefficients typically have no effect on the supports of the basis functions, but do have a profound effect on the shape of the basis functions themselves, and hence on the numerical values appearing in the hierarchical basis stiffness matrix [BX94], [BX96]. One can even choose different coefficients for the trial and test spaces, leading to different hierarchical basis functions, similar to Petrov-Galerkin finite element approximations.

In terms of ILU, the expansion coefficients are just the multipliers in the ILU decomposition. This leads one to consider the possibility of defining these expansion coefficients in a more algebraic fashion. For example, one can choose these coefficients to eliminate certain off diagonal elements of the hierarchical basis stiffness matrix, as is done in the case of classical ILU. While geometry based coefficients seem adequate for isotropic, self-adjoint problems (e.g. $-\Delta u = f$), we have found that our algebraic choices can greatly improve the robustness of the HBMG iteration for other types of equations, notably convection dominated convection-diffusion equations.

In Section 2, we review the connection between HBMG and ILU from a graph theoretical point of view. In Section 3, we analyze this method in the one-dimensional case, where algebraic simplicity allows a fairly complete treatment. In Section 4, we present some algorithms and numerical results for two-dimensional problems.

2 HBMG and ILU

As with typical multigrid methods, classical hierarchical basis methods are usually defined in terms of an underlying refinement structure for a sequence of nested meshes. In many cases, this is no disadvantage, but it limits the applicability of the methods to truly unstructured meshes, which might be highly nonuniform but not derived from some grid refinement process. Here we view the transformation of the stiffness matrix A from nodal basis representation to hierarchical basis representation as a special ILU decomposition. This generalizes the construction of hierarchical bases to unstructured meshes, allowing HBMG and other hierarchical basis methods to be applied. A more complete discussion of this point can be found in [BX94] and [BX96]. See [HS95], [Kor96], [HK94], [KY94], and [CS94] for related algorithms. See [Xu89], [BPX91], and [Zha88] for discussions of non-nested multigrid algorithms.

Graph Theoretical Properties of Hierarchical Bases

We begin by exploring the connection between the HBMG method and ILU decomposition in terms of graph theory. We consider the standard Gaussian elimination and the classical ILU factorization from a graph theoretical point of view, and then develop a simple graph elimination model for classical hierarchical basis methods on sequences of nested meshes. We can interpret this model as a particular ILU decomposition and generalize this graph elimination model to the case of completely unstructured meshes. See Rose [Ros72] and George and Liu [GL81] for a complete discussion of graph theoretical aspects of Gaussian elimination. Corresponding to a sparse $n \times n$ matrix A with symmetric pattern (i.e. $A_{ij} \neq 0$ if and only if $A_{ji} \neq 0$), let $G(X, E)$ be the graph that consists of a set of n ordered vertices $v_i \in X$, $1 \leq i \leq n$, and a set of edges E such that the edge (connecting vertices v_i and v_j) $e_{ij} \in E$ if and only if $a_{ij} \neq 0$, $i \neq j$. The edges in the graph G correspond to the nonzero off diagonal entries of A. If A is the stiffness matrix for the space of continuous piecewise linear polynomials represented in the standard nodal basis, the graph G is just the underlying triangulation of the domain (with minor modifications due to Dirichlet boundary conditions). We define for a vertex v_i the set of adjacent vertices $adj(v_i)$ by

$$adj(v_i) = \{v_j \in X | e_{ij} \in E\}.$$

A clique $C \subseteq X$ is a set of vertices which are all pairwise connected; that is $v_i, v_j \in C, i \neq j \Rightarrow e_{ij} \in E$. With a proper ordering of the vertices, a clique corresponds to a dense submatrix of A. In graph theoretic terms, a single step of Gaussian elimination transforms $G(X, E)$ to a new graph $G'(X', E')$ as follows:

1. Eliminate vertex v_i and all its incident edges from G. Set $X' = X - \{v_i\}$. Denote the resulting set of edges $E_1 \subseteq E$.
2. Create a set F of fillin edges as follows: For each distinct pair $v_j, v_k \in adj(v_i)$ in G, add the edge e_{jk} to F if not already present in E_1. Set $E' = E_1 \cup F$.

Note that the set $adj(v)$ in G becomes a clique in G'. Within this framework, the classical ILU factorization is one in which *no* fillin edges are allowed, i.e. $F \equiv \emptyset$. This forces the matrix A' corresponding to the new graph G' to have the same sparsity structure as the corresponding submatrix of A.

To define HBMG as a generalized ILU procedure, we must first introduce the concept of *vertex parents.* We will begin with the case of two nested meshes where the fine mesh is a uniform refinement of a coarse mesh, generated by pairwise connecting the midpoints of the coarse grid edges in the usual way [BDY88], [Yse86], [Hac85]. Here we can make the direct sum decomposition $X = X_c \oplus X_f$, where X_c is the set of coarse grid vertices and X_f is the set of fine grid vertices (those not in X_c). For each vertex $v_i \in X_f$, there is a unique pair of vertex parents $v_j, v_k \in X_c$ such that v_i is the midpoint of the edge connecting v_j and v_k ($v_i = (v_j + v_k)/2$).

We now view HBMG as an ILU algorithm in which only selected fillin edges are allowed, namely those connecting vertex parents. In this algorithm, we sequentially eliminate the vertices in the set X_f as follows:

1. Eliminate vertex $v_i \in X_f$ and all its incident edges from G. Set $X' = X - \{v_i\}$. Denote the resulting set of edges $E_1 \subseteq E$.
2. Add one fillin edge connecting the vertex parents $v_j, v_k \in X_c$ of v_i. Set $E' = E_1 \cup \{e_{jk}\}$.

Note that the triangulation $\mathcal{T}_f$ is the graph for the original stiffness matrix A represented in the standard nodal basis. After all the vertices in X_f are eliminated, the resulting graph is just the triangulation $\mathcal{T}_c$, i.e. the sparsity pattern of the coarse grid matrix corresponds to the coarse grid triangulation. For completely unstructured meshes, the main problem is to determine reasonable vertex parents for each vertex to be eliminated. Once this is done, the elimination/unrefinement/coarsening is done exactly as in the case of nested meshes. This may lead to graphs that are not necessarily triangulations of the domain, but typically contain polygonal elements of various orders. Even if the graphs remain triangulations, they will generally not be nested. Nonetheless, such a scheme still defines the linear combinations of fine grid basis functions used to create the coarse grid basis functions (but not the values of the coefficients in the linear combinations).

Algorithms for selecting vertex parents are currently an area of active research. The scheme developed in [BX96] is based on the geometry of the triangulation, and seeks to coarsen the grid in a fashion that maintains the shape of the region and the shape regularity of the coarse grid elements to the extent possible. For this scheme, the supports of the resulting hierarchical basis functions grow in a fashion analogous to the classical case. We are also considering more algebraic schemes which rely only on the sparsity structure of the stiffness matrix and the numerical values of its matrix elements; these schemes have much in common with more classical sparse matrix ordering algorithms.

Algebraic HBMG and ILU

In this section, we consider the algebraic aspects of the HBMG method and its relation to Gaussian elimination. Again we will consider the case of only two levels. Let A denote the nodal basis stiffness matrix for the fine grid, and consider the block partitioning

$$A = \begin{pmatrix} A_{cc} & A_{cf} \\ A_{fc} & A_{ff} \end{pmatrix}, \tag{1}$$

where A_{ff} corresponds to the nodal basis functions of the fine grid nodes, A_{cc} corresponds to the (fine grid) nodal basis functions of the coarse grid nodes and A_{cf} and A_{fc} correspond to the coupling between the two sets of basis functions. We consider transformations of the form $A' = S^T A\tilde{S}$ where S and $\tilde{S}$ have the block structure

$$S = \begin{pmatrix} I & 0 \\ R & I \end{pmatrix}, \quad \tilde{S} = \begin{pmatrix} I & 0 \\ \tilde{R} & I \end{pmatrix}. \tag{2}$$

By direct calculation, we obtain

$$S^T A\tilde{S} = \begin{pmatrix} \hat{A}_{cc} & A_{cf} + R^T A_{ff} \\ A_{fc} + A_{ff}\tilde{R} & A_{ff} \end{pmatrix} \tag{3}$$

where

$$\hat{A}_{cc} = A_{cc} + R^T A_{fc} + A_{cf}\tilde{R} + R^T A_{ff}\tilde{R}. \tag{4}$$

Different algorithms can be characterized by different choices of R and $\tilde{R}$. For example, in the classical block Gaussian elimination we have $R = -A_{ff}^{-T} A_{cf}^T$ and $\tilde{R} = -A_{ff}^{-1} A_{fc}$, and $\hat{A}_{cc} = A_{cc} - A_{cf} A_{ff}^{-1} A_{fc}$ is the Schur complement. In this case, the off diagonal blocks are reduced to zero, but at the cost of having fairly dense matrices R, $\tilde{R}$ and $\hat{A}_{cc}$. In the case of HBMG, the matrices R and $\tilde{R}$ are sparse and contain information about changing from the nodal to the hierarchical basis. The sparsity patterns of R and $\tilde{R}$ are the same, and both are determined by the vertex parent relationship described above. In particular, each row of R and $\tilde{R}$ is zero except for the two entries which correspond to the (coarse grid) vertex parents for the given fine grid vertex. In the classical case, where each fine grid vertex is the midpoint of the edge connecting its vertex parents, $R = \tilde{R}$ and both nonzero entries in a given row are equal to $1/2$. In the generalized HBMG, these values are replaced by $\theta_i, \tilde{\theta}_i, \nu_i$ and $\tilde{\nu}_i$. Often one has $\tilde{\theta}_i = 1 - \theta_i$ and $\tilde{\nu}_i = 1 - \nu_i$. Choosing $R \neq \tilde{R}$ corresponds to choosing a test space different from the trial space for the coarser grids, as in a Petrov-Galerkin method. Several alternatives for choosing $\theta_i, \tilde{\theta}_i, \nu_i$ and $\tilde{\nu}_i$ are discussed in Section 4.

3 Analysis of a One-Dimensional Model Problem

The Two-Level HBMG Iteration

In this section, we will analyze the case of a constant coefficient two two-point boundary value problem, giving rise to a constant coefficient tridiagonal stiffness matrix when discretized using some finite element approximation on a uniform mesh. See [Hac84] and [BB91] for other analyses of multilevel methods for one-dimensional model problems. Let $n > 2$ be an integer and set $h = 1/(2n)$. The uniform fine mesh $\mathcal{T}_h$ of size h has $2n + 1$ grid points $x_k = kh, 0 \leq k \leq 2n$. The coarse mesh has $n + 1$ grid points $x_{2k}, 0 \leq k \leq n$. We will refer to the set of coarse grid points as level 1 vertices, and $x_{2k+1}, 0 \leq k \leq n - 1$, as level 2 vertices.

Let $\mathcal{P}_h$ be a $2n - 1$ dimensional trial space of functions associated with the fine mesh $\mathcal{T}_h$ satisfying the boundary conditions $v(x_0) = v(x_{2n}) = 0$ for all $v \in \mathcal{P}_h$. Let $\hat{\phi}_k, 1 \leq k \leq 2n - 1$, denote the nodal basis for the trial space $\mathcal{P}_h$. We assume that $support\{\hat{\phi}_k\} = (x_{k-1}, x_{k+1})$ and that $\hat{\phi}_k(x_j) = \delta_{kj}$. Similarly, we define the $2n - 1$ dimensional test space $\mathcal{S}_h$ with nodal basis $\hat{\psi}_k, 1 \leq k \leq 2n - 1$.

We will use a discretization on the mesh $\mathcal{T}_h$ given by a bilinear form $b(\cdot, \cdot) : \mathcal{P}_h \times \mathcal{S}_h \rightarrow \mathbb{R}$. The details of the bilinear form $b(\cdot, \cdot)$ are arbitrary for the moment. The discrete system of equations to be solved is: Find $u_h \in \mathcal{P}_h$ such that

$$b(u_h, v) = rhs(v)$$

for all $v \in \mathcal{S}_h$, where $rhs(\cdot)$ is an appropriate linear functional.

Suppose

$$b(\hat{\phi}_k, \hat{\psi}_j) = \begin{cases} a & j = k - 1 \\ b & j = k + 1 \\ c & j = k \\ 0 & |k - j| > 1 \end{cases} \tag{5}$$

where a, b, and c are constants.

The resulting nodal basis stiffness matrix in natural vertex order is the constant coefficient tridiagonal matrix

$$A^{NB} = \begin{pmatrix} c & b & & & \\ a & c & b & & \\ & \ddots & \ddots & \ddots & \\ & & a & c & b \\ & & & a & c \end{pmatrix}. \tag{6}$$

Now we introduce the generalized hierarchical bases for the spaces $\mathcal{P}_h$ and $\mathcal{S}_h$. Define functions

$$\tilde{\phi}_{2k} = \theta\hat{\phi}_{2k-1} + \hat{\phi}_{2k} + \tilde{\theta}\hat{\phi}_{2k+1} \tag{7}$$

for $1 \le k \le n-1$, and θ, $\tilde{\theta} \in \mathbb{R}$. The generalized hierarchical basis for $\mathcal{P}_h$ consists of the union of the functions $\tilde{\phi}_{2k}$ for $1 \le k \le n-1$, and the basis functions for the level 2 nodes, $\hat{\phi}_{2k+1}$, $0 \le k \le n-1$. This basis will be denoted by ϕ_k, $1 \le k \le 2n-1$. The generalized hierarchical basis introduces a natural direct sum decomposition of the space $\mathcal{P}_h$. If $u \in \mathcal{P}_h$, then we have the unique decomposition $u = v + w$, where $v \in \mathcal{V} = span\{\hat{\phi}_{2k+1}\}_{k=0}^{n-1}$ and $w \in \mathcal{W} = span\{\tilde{\phi}_{2k}\}_{k=1}^{n-1}$. The generalized hierarchical basis $\psi_k, 1 \le k \le 2n-1$, of $\mathcal{S}_h$ is defined similarly, but with constants ν and $\tilde{\nu}$ instead of θ and $\tilde{\theta}$.

Using (5) and (7), we obtain

$$b(\phi_{2k}, \psi_j) = \begin{cases} \hat{a} = \theta\tilde{\nu}c + (\tilde{\nu} + \theta)a & j = 2k-2 \\ p = \theta c + a & j = 2k-1 \\ \hat{c} = (1 + \tilde{\theta}\tilde{\nu} + \theta\nu)c + (\theta + \tilde{\nu})b + (\nu + \tilde{\theta})a & j = 2k \\ q = \tilde{\theta}c + b & j = 2k+1 \\ \hat{b} = \nu\tilde{\theta}c + (\nu + \tilde{\theta})b & j = 2k+2 \end{cases} \tag{8}$$

for $1 \le k \le n-1$, and

$$b(\phi_{2k+1}, \psi_j) = \begin{cases} r = \tilde{\nu}c + a & j = 2k \\ c & j = 2k+1 \\ s = \nu c + b & j = 2k+2 \end{cases} \tag{9}$$

for $0 \le k \le n-1$.

The stiffness matrix A^{HB} corresponding to the hierarchical basis is a pentadiagonal matrix given by

$$A^{HB} = \begin{pmatrix} c & s & 0 & & & & & \\ p & \hat{c} & q & \hat{b} & & & & \\ 0 & r & c & s & 0 & & & \\ & \hat{a} & p & \hat{c} & q & \hat{b} & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & 0 & r & c & s & 0 \\ & & & & \hat{a} & p & \hat{c} & q \\ & & & & & 0 & r & c \end{pmatrix}.$$

Now we apply a simple permutation to the matrix A^{HB}, in which the basis functions associated with the coarse grid points are ordered first, and those associated with the fine grid points are ordered last. If we denote the relevant permutation matrix by P, the permuted matrix is block 2×2 of the form given in (3)-(4):

$$\bar{A}^{HB} = PA^{HB}P^T = \begin{pmatrix} A_{cc} & A_{cf} \\ A_{fc} & A_{ff} \end{pmatrix},$$

where

$$A_{cc} = \begin{pmatrix} \hat{c} & \hat{b} & & & \\ \hat{a} & \hat{c} & \hat{b} & & \\ & \ddots & \ddots & \ddots & \\ & & \hat{a} & \hat{c} & \hat{b} \\ & & & \hat{a} & \hat{c} \end{pmatrix}_{n-1\times n-1}, \quad A_{cf} = \begin{pmatrix} p & q & & & \\ & p & q & & \\ & & \ddots & \ddots & \\ & & & p & q \end{pmatrix}_{n-1\times n},$$

$$A_{fc} = \begin{pmatrix} r & & & \\ s & r & & \\ & s & \ddots & \\ & & \ddots & r \\ & & & s \end{pmatrix}_{n\times n-1}, \quad A_{ff} = \begin{pmatrix} c & & & \\ & c & & \\ & & \ddots & \\ & & & c \end{pmatrix}_{n\times n}.$$

The 2-level generalized hierarchical basis multigrid method is the block symmetric Gauß-Seidel iteration applied to the linear system $\bar{A}^{HB}\bar{u} = \bar{f}$. If $\bar{A}^{HB}$ is block upper (or lower) triangular, we will obtain the exact solution in one step. To make $\bar{A}^{HB}$ block diagonal, we set $p = q = r = s = 0$, giving

$$\theta = \tilde{\nu} = -\frac{a}{c} \quad \text{and} \quad \tilde{\theta} = \nu = -\frac{b}{c}. \tag{10}$$

For this choice of interpolation coefficients, the transformation of the stiffness matrix $\bar{A}^{NB}$ to the matrix $\bar{A}^{HB} = S^T\bar{A}^{NB}\tilde{S}$ is the classical block Gaussian elimination.

Examples

For our first example, we consider the self adjoint problem $-u'' = f$ with Dirichlet boundary conditions. Discretizing using the standard nodal basis for the space of continuous piecewise linear polynomials leads to the tridiagonal stiffness matrix

$$A^{NB} = \frac{1}{h}\begin{pmatrix} 2 & -1 & \\ -1 & \ddots & \ddots \\ & \ddots & \end{pmatrix}.$$

If we use the standard interpolation constants,

$$\theta = \tilde{\theta} = \nu = \tilde{\nu} = \frac{1}{2},$$

leading to the standard piecewise linear nodal basis functions for the course grid, the permuted hierarchical basis stiffness matrix is of the form

$$\bar{A}^{HB} = \frac{1}{2h}\left(\begin{array}{cccccc} 2 & -1 & & | & & \\ -1 & \ddots & \ddots & | & & \\ & \ddots & & | & & \\ - & - & - & - & - & - \\ & & & | & 4 & \\ & & & | & & \ddots \end{array}\right).$$

It is well known that the HBMG method is a direct method for this special case [Yse86].

As our second example, we consider the one-dimensional convection diffusion equation $-(u' + \beta u)' = f$ on an interval I, with Dirichlet boundary conditions and constant β. Here we will use the well known Scharfetter-Gummel discretization on a uniform mesh. There are several standard interpretations of this discretization. One which is especially useful here is to view the discretization as a Petrov-Galerkin method using the standard bilinear form

$$b(u, v) = \int_I (u' + \beta u)v' \, dx.$$

For the test space we use the standard continuous piecewise linear polynomials, while the trial space is composed of piecewise functions of the form $\alpha e^{-\beta x} + \gamma$, since these are the fundamental solutions of the homogeneous equation. The nodal basis function $\phi_i(x)$ is given by

$$\phi_i(x) = \begin{cases} (e^{-\beta(x-x_i)} - e^{-\beta h})/(1 - e^{-\beta h}) & x_i \leq x \leq x_{i+1} \\ (e^{-\beta(x-x_i)} - e^{\beta h})/(1 - e^{\beta h}) & x_{i-1} \leq x \leq x_i \\ 0 & \text{elsewhere} \end{cases}$$

The resulting tridiagonal nodal basis stiffness matrix has entries

$$\begin{aligned} a &= -\frac{\mathcal{B}(-\beta h)}{h}, \\ b &= -\frac{\mathcal{B}(\beta h)}{h}, \\ c &= \frac{\mathcal{B}(\beta h) + \mathcal{B}(-\beta h)}{h}, \end{aligned}$$

where $\mathcal{B}(\cdot)$ denotes the Bernoulli function

$$\mathcal{B}(x) = \frac{x}{e^x - 1}.$$

To make the off-diagonal blocks zero, we choose interpolation coefficients

$$\begin{aligned} \theta &= \tilde{\nu} = \frac{\mathcal{B}(-\beta h)}{\mathcal{B}(-\beta h) + \mathcal{B}(\beta h)} = \frac{1 - e^{-\beta h}}{1 - e^{-2\beta h}}, \\ \tilde{\theta} &= \nu = 1 - \theta. \end{aligned}$$

The entries of the resulting coarse grid matrix are given by

$$\begin{aligned}\hat{a} &= -\frac{\mathcal{B}(-2\beta h)}{2h},\\ \hat{b} &= -\frac{\mathcal{B}(2\beta h)}{2h},\\ \hat{c} &= \frac{\mathcal{B}(2\beta h)+\mathcal{B}(-2\beta h)}{2h},\end{aligned}$$

i.e., with this choice of interpolation coefficients the Scharfetter-Gummel discretization on the coarse grid is reproduced. This is the same result we get calculating θ from $u(x+h/2)=\theta u(x)+(1-\theta)u(x+h)$ and making the interpolation exact for functions of the form $u(x)=\alpha e^{-\beta x}+\gamma$.

For our third example, we consider the case of the L^2-projection into the space of continuous piecewise linear finite elements. The mass matrix for the space of continuous piecewise linear polynomials on a uniform mesh is given by

$$M=\frac{h}{6}\begin{pmatrix} 4 & 1 & \\ 1 & \ddots & \ddots \\ & \ddots & \end{pmatrix}.$$

In order to make the off-diagonal blocks zero, we have to choose

$$\theta=\tilde{\theta}=\nu=\tilde{\nu}=-\frac{1}{4}.$$

Here the coefficients are negative and $\theta+\tilde{\theta}\neq 1$. Note that this leads to oscillatory wavelet like basis functions in the k-level recursion.

4 Some Interpolation Algorithms for Two Dimensions

In this section, we consider the two-dimensional convection-diffusion equation

$$-\nabla(\nabla u+\beta u)=f \quad \text{in } \Omega \tag{11}$$

with boundary conditions

$$u=0 \quad \text{on } \partial\Omega, \tag{12}$$

where Ω is a polygonal domain and β is constant. We will apply the Scharfetter Gummel discretization on an unstructured triangular mesh. We will discuss several alternatives for choosing interpolation coefficients $\theta, \tilde{\theta}, \nu$ and $\tilde{\nu}$ which determine the generalized hierarchical basis functions. Unfortunately, in 2-D it is not possible to make the off-diagonal blocks equal to zero. Most alternatives are motivated by the one-dimensional case examined in Section 3.

Classical HBMG (Linear Interpolation)

In the classical HBMG method, the interpolation coefficients are chosen to exactly interpolate one-dimensional linear polynomials along element edges. The hierarchical basis stiffness matrix is of the form $A^{HB} = S^T A^{NB} S$ as given in (3)-(4). As usual, the matrices R and $\tilde{R}$ of (2) contain the interpolation coefficients, which in the case of a regular uniform refinement satisfy $\theta = \tilde{\theta} = \nu = \tilde{\nu} = 1/2$. If a fine grid point v_i is not the midpoint of its vertex parents v_j, v_k, we take the corresponding fractions $\theta = \nu = dist(v_i, v_k)/dist(v_j, v_k)$ and $\tilde{\theta} = \tilde{\nu} = 1 - \theta$.

The Scharfetter-Gummel method (Exponential Interpolation)

An exponential interpolation scheme can be derived from the Scharfetter-Gummel formula. We begin by noting that, analogous to the one-dimensional case, fundamental solutions of (11) are given by

$$u(x) = \alpha + \gamma e^{-\langle\beta,x\rangle} = \alpha + \gamma e^{-\beta_1 x_1 - \beta_2 x_2}$$

for constants $\alpha, \gamma \in \mathbb{R}$. Suppose the values $u_1 = u(v_1)$ and $u_2 = u(v_2)$ are known and $u_m \equiv u(v_m) = u(\theta v_1 + (1-\theta)v_2)$ is to be approximated. If we require an exact interpolation of the fundamental solutions on the one-dimensional edge between v_1 and v_2, we can obtain by a straightforward calculation $u_m = \nu u_1 + \tilde{\nu} u_2$, where

$$\begin{aligned} \nu &= \frac{\theta \mathcal{B}(\langle\beta, v_2 - v_1\rangle)}{\mathcal{B}(\theta\langle\beta, v_2 - v_1\rangle)} = \frac{e^{\theta\langle\beta, v_2 - v_1\rangle} - 1}{e^{\langle\beta, v_2 - v_1\rangle} - 1}, \\ \tilde{\nu} &= 1 - \nu. \end{aligned}$$

Here $\mathcal{B}(x)$ denotes the Bernoulli function. When $\beta = 0$, this method reduces to the classical HBMG algorithm. Note that the interpolation coefficients lie in $(0,1)$ and sum up to 1: $\nu + \tilde{\nu} = 1$. We note that problems arise for multilevel methods where the algebraic coarse grid matrices do not correspond to a discretized convection-diffusion equation anymore. In some very special cases, with a slight variation of the SG-coefficients, one can force A_{cc}^{HB} to be the coarse grid Scharfetter-Gummel discretization matrix. However, this leads to poor numerical results.

An Algebraic ILU Method

In this method, we choose interpolation coefficients to create zeroes in the off-diagonal blocks whenever possible, in a fashion analogous to Gaussian elimination. This leads to coefficients $\theta_k^i = -a_{ik}/a_{ii}$, where vertex i is the fine grid vertex to be eliminated and vertex k is one of its vertex parents. If vertex i is the only one to be eliminated, this leads to a minimization of the $\|\cdot\|_1$ and $\|\cdot\|_2$ norms of the vectors $A_{cf} + R^T a_{ff}$ and $A_{fc} + a_{ff}\tilde{R}$. In the general case, the corresponding norms of the affected row and column vectors are locally minimized at each elimination step. The interpolation coefficients can have either sign, and generally $\theta + \tilde{\theta} \neq 1$. A related possibility is to choose $\theta_j^i = a_{ij}/(a_{ij} + a_{ji})$, for which the coefficients will sum to one. Interestingly, for the case of a uniform mesh of isosceles right triangles with regular refinement, the coefficients in $x-$ and $y-$direction are the Scharfetter-Gummel coefficients, but the "diagonal" interpolation coefficients are indeterminate and could for example be taken as zero.

Minimizing the Frobenius Norm

In this method the interpolation coefficients are chosen such that the Frobenius norm of the off-diagonal blocks is minimized. Intuitively it appears that making off-diagonal blocks smaller should increase the rate of convergence, since these are the blocks which are lagged in the iteration. Thus minimizing some norm of the off-diagonal blocks might be good. On the other hand, in this case the minimization leads to small sets of linear equations to be solved for the interpolation coefficients. These linear systems are awkward to assemble if the stiffness matrix is represented in some of the standard sparse matrix formats, and the solution process adds to the cost of the method. In our experience with the method, we noted no significant improvement in the convergence properties of the resulting hierarchical basis iteration. Thus, at this time we cannot recommend this approach.

Hybrid Methods

We have also considered combinations of the above methods for the trial and test space, effectively making a Petrov-Galerkin like method for the coarse grid approximation. Note from (3) that $A^{HB}_{fc} = A^{NB}_{fc} + A^{NB}_{ff}\tilde{R}$ is influenced only by $\tilde{R}$ and $A^{HB}_{cf} = A^{NB}_{cf} + R^T A^{NB}_{ff}$ is influenced only by R. The motivation is that one needs A^{HB}_{cf} *or* A^{HB}_{fc} to be small in order to get good convergence for the symmetric Gauß-Seidel iteration. Thus, one could use interpolation coefficients for one space to make the corresponding off-diagonal block small, and choose those for the other space to influence A^{HB}_{cc} in order to get favorable recurrence relations for several levels. Note that this can lead to a non-symmetric matrix A^{HB} even if A^{NB} is symmetric. One possibility we have found to be effective is to use linear interpolation coefficients for the trial space and algebraic ILU coefficients for the test space.

Numerical Results

In this section, we present numerical illustrations for some of the interpolation schemes applied to the model convection-diffusion equation (11)-(12), with $f \equiv 1$ and $\Omega \equiv (0,1) \times (0,1)$. The problem is discretized using the Scharfetter-Gummel method. The level 1 mesh is a uniform 5×5 mesh shown in Figure 1 (note boundary vertices do not correspond to unknowns). This mesh is uniformly refined by dividing each triangle into four congruent triangles using regular refinement. The refinement is continued until we reach level 5 (6) with 4225 (16641) vertices. We used uniform grids with structured refinement rather than the symbolic elimination for unstructured grids as described in Section 2 in order to treat all test cases in a more standardized setting. To show the performance of the methods on unstructured grids, we adaptively refined the level 5 grid until we reached 10000 vertices. This results in 7 levels of refinement.

We illustrate the dependence of the convergence rate on the direction and magnitude of β. The results for the different methods are shown in Table 1 for structured grids and in Table 2 for unstructured grids. We accelerated the iteration with the bicg

Table 1 Average convergence rates for various methods on structured grids and for several values of $\beta = (\beta_1, \beta_2)^T$. "Lin", "SG" and "ILU" denote linear, exponential, and ILU interpolation coefficients, respectively. For example, "ILU, Lin" means ILU interpolation coefficients were used for the trial space, and linear interpolation was used for the test space.

$N = 4225$	Lin, Lin	SG, SG	SG, Lin	ILU, ILU	ILU, Lin
$(0, 0)$	0.45	0.45	0.45	0.73	0.42
$(0, 1000)$	fails	0.88	0.73	0.49	0.50
$(0, 5000)$	fails	0.85	0.87	0.67	0.61
$(707, 707)$	fails	0.87	0.61	0.70	0.48
$(3536, 3536)$	fails	fails	0.61	0.70	0.48
$(-707, 707)$	fails	0.85	0.75	0.67	0.54
$(-3536, 3536)$	fails	0.85	0.75	0.67	0.54
$N = 16641$	Lin, Lin	SG, SG	SG, Lin	ILU, ILU	ILU, Lin
$(0, 0)$	0.47	0.47	0.47	0.85	0.52
$(0, 1000)$	fails	0.93	0.84	0.76	0.50
$(0, 5000)$	fails	0.94	0.92	0.71	0.61
$(707, 707)$	fails	fails	0.71	0.94	0.61
$(3536, 3536)$	fails	fails	0.71	0.92	0.60
$(-707, 707)$	fails	0.94	0.84	0.97	0.68
$(-3536, 3536)$	fails	0.94	0.88	0.92	0.68

Table 2 Average convergence rates for various methods on unstructured grids and for several values of $\beta = (\beta_1, \beta_2)^T$. "Lin", "SG" and "ILU" denote linear, exponential, and ILU interpolation coefficients, respectively. For example, "ILU, Lin" means ILU interpolation coefficients were used for the trial space, and linear interpolation was used for the test space.

$N = 10000$	Lin, Lin	SG, SG	SG, Lin	ILU, ILU	ILU, Lin
$(0, 0)$	0.44	0.44	0.44	0.84	0.34
$(0, 1000)$	fails	0.44	0.78	0.69	0.56
$(0, 5000)$	fails	0.92	0.90	0.76	0.71
$(707, 707)$	fails	0.95	0.70	0.73	0.53
$(3536, 3536)$	fails	0.94	0.70	0.75	0.57
$(-707, 707)$	fails	0.93	0.76	0.80	0.61
$(-3536, 3536)$	fails	0.92	0.74	0.82	0.60

method. We record average rates of convergence after $k = \min\{100, \bar{k}\}$ iterations, where the residual is reduced by 10^{-2} in $\bar{k}$ steps. The average rate of convergence is given by $\gamma = (\|r_k\|/\|r_0\|)^{\frac{1}{k}}$, where r_i denotes the residual after i steps. We choose $x_0 = (0, 0, \cdots, 0)^T$ as starting vector for the purpose of standardization. All calculations were done in double precision on a Sparc10/41.

Figure 1 The initial mesh (left) and convergence histories for N=16641, ILU/ILU, $\beta = (-3536, 3536)^T$ (middle) and ILU/Lin, $\beta = (-3536, 3536)^T$ (right).

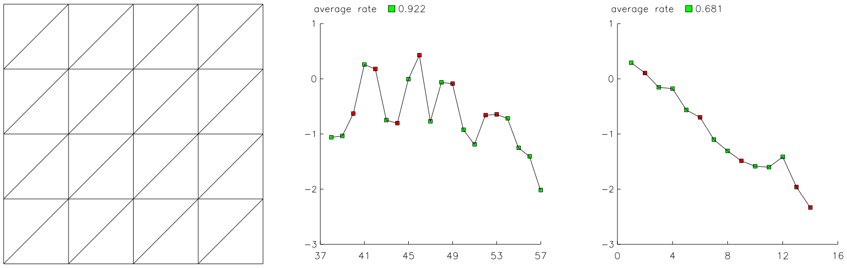
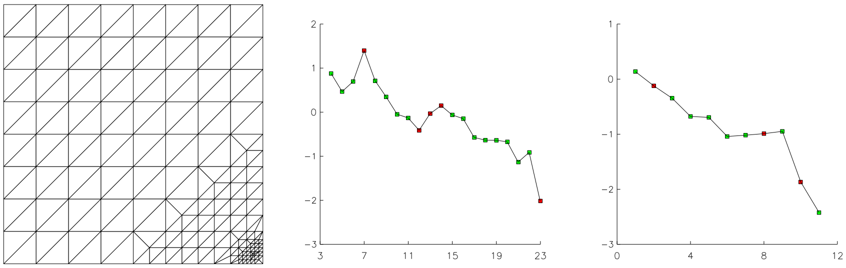


Figure 2 An unstructured mesh (left) and convergence histories for N=10000, ILU/ILU, $\beta = (-3536, 3536)^T$ (middle) and ILU/Lin, $\beta = (-3536, 3536)^T$ (right).



Typical convergence histories are shown in Figure 1 for a structured grid and in Figure 2 for an adaptively refined grid, where $\log \|r_i\|/\|r_0\|$ is plotted as a function of the iteration index i. Here we observe the non-monotonic behavior of the residual typical of the bicg method for strongly nonsymmetric problems. Iterations which failed did not reduce the initial residual in 100 iterations, but might have succeeded with more iterations. The coarse grid in these experiments is extremely coarse for such problems. As is standard with multilevel methods, one can overcome convergence failure and/or improve the rates of convergence by making the coarse grid finer. However, our intent here is only to illustrate that our alternative interpolation schemes improve the robustness of the HBMG iteration, often allowing rapid convergence even under very adverse conditions.

Acknowledgement

The work of the first author was supported by the U. S. Office of Naval Research under contract N00014-89J-1440. The second author was supported by a DAAD-fellowship HSPII from the German Federal Ministry for Education, Science, Research and Technology.

REFERENCES

[BB91] Bank R. E. and Benbourenane M. (1991) A Fourier analysis of the two level hierarchical basis multigrid method for convection diffusion equations. In *Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations (R. Glowinski, Y. Kuznetsov, G. Meurant, J. Périaux, and O. Widlund, eds.)*, pages 178–184. SIAM, Philadelphia.

[BDY88] Bank R. E., Dupont T. F., and Yserentant H. (1988) The hierarchical basis multigrid method. *Numer. Math.* 52: 427–458.

[BPX91] Bramble J., Pasciak J., and Xu J. (1991) The analysis of multigrid algorithms with non-imbedded spaces or non-inherited quadratic forms. *Math. Comp.* 56: 1–43.

[BX94] Bank R. E. and Xu J. (1994) The hierarchical basis multigrid method and incomplete LU decomposition. In *Seventh International Symposium on Domain Decomposition Methods for Partial Differential Equations (D. Keyes and J. Xu, eds.)*, pages 163–173. AMS, Providence, Rhode Island.

[BX96] Bank R. E. and Xu J. (1996) An algorithm for coarsening unstructured meshes. *Numerische Mathematik* 73: 1–36.

[CS94] Chan T. F. and Smith B. F. (1994) Domain decomposition and multigrid algorithms for elliptic problems on unstructured meshes. In *Proceedings of Seventh International Conference on Domain Decomposition. (ed. D. Keyes and J. Xu)*, pages 175–189. AMS, Providence, Rhode Island.

[GL81] George A. and Liu J. (1981) *Computer Solution of Large Sparse Positive Definite Systems.* Prentice Hall, Englewood Cliffs, NJ.

[Hac84] Hackbusch W. (1984) Multigrid convergence for a singular perturbation problem. *Lin. Alge. Appl.* 58: 125–145.

[Hac85] Hackbusch W. (1985) *Multigrid Methods and Applications.* Springer-Verlag, Berlin.

[HK94] Hoppe R. H. W. and Kornhuber R. (1994) Adaptive multilevel methods for obstacle problems. *SIAM J. Numer. Anal.* 31: 301–323.

[HS95] Hackbusch W. and Sauter S. A. (1995) A new finite element space for the approximation of pdes on domains with complicated microstructure. Technical report, Universität Kiel.

[Kor96] Kornhuber R. (1996) Monotone multigrid methods for variational inequalities I. *Numer. Math.* 72: 49–60.

[KY94] Kornhuber R. and Yserentant H. (1994) Multilevel methods for elliptic problems of domains not resolved by the coarse grid. In *Seventh International Symposium on Domain Decomposition Methods for Partial Differential Equations (D. Keyes and J. Xu, eds.)*, pages 49–60. AMS, Providence, Rhode Island.

[Ros72] Rose D. J. (1972) A graph theoretic study of the numeric solution of sparse positive definite systems. In *Graph Theory and Computing.* Academic Press, New York.

[Xu89] Xu J. (1989) *Theory of Multilevel Methods.* PhD thesis, Cornell University. Report AM-48, Penn State.

[Yse86] Yserentant H. (1986) On the multi-level splitting of finite element spaces. *Numer. Math.* 49: 379–412.

[Zha88] Zhang S. (1988) *Multilevel Iterative Techniques.* PhD thesis, Pennsylvania

State University, Department of Mathematics Report 88020.

31

A Domain Decomposition Method for Control Problems

Jean-David Benamou

1 Introduction

A DDM (Domain Decomposition Method) for the optimal control of systems governed by PDEs (Partial Differential Equations) is presented. The general framework is based on a nonoverlapping spatial decomposition of the domain (time is not decomposed for evolution equations) and the introduction of *skew-symmetric, Robin, iterative transmission conditions* between subdomains which couple the direct and adjoint states appearing in the optimality system (derived from the control problem). A reference paper on this family of DDM is [Lio90]; see also [RG90b, Des91, Des93, BD96] on the extension of this method to the Helmholtz equation.

Because of the natural coupling between direct and adjoint states in optimality systems, a general strategy for proving convergence of the DDM is available which is independent of the nature of the governing equation. In our opinion, this general convergence property makes the SRC (Skew-symmetric, Robin, Coupled) transmissions conditions the natural choice to domain decompose control problems.

Unlike for PDEs, the resolution of optimal control problem using DDMs has received little attention (at least in the previous edition of this conference). On this precise subject we can only cite [AB73, Bou, Leu96]. A multigrid method can also be found in [W.H79]. Any DDM, relevant to solve a PDE, can of course be used as the kernel of an optimization algorithm (of gradient type for instance). The originality of our approach, developed in [Ben93, Ben96a, Ben96b, BD96, Ben95b, Ben95a], consists in decomposing the full optimality system. This means that our DDM solves concurrently the equations and the optimization problem.

In the following two sections we briefly present the method on simple model problems. We then describe a general proof of convergence. We discuss, in a fourth section, the possible extension of this method to more complicated and also different control problems. Based on our own experiments, we finally give a few remarks on the implementation of this method.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 The Model Problems

The goal of the control problem is to minimize a cost function:

$$\min_{u\in U} J(u, y(u)) \tag{1}$$

where u is a an admissible control which acts on $y(u)$ the solution of a PDE which can be either elliptic, parabolic or hyperbolic (to avoid unnecessary complications in the notations, we drop the dependence of y in u in the remainder of the paper):

$$\begin{aligned}
&-\Delta y(x) = f(x) + u(x) \quad \text{on} \quad \Omega\\
&(\frac{\partial}{\partial t} - \Delta)y(t,x) = f(t,x) + u(t,x) \quad \text{on} \quad]0,T[\times\Omega, \text{ with initial condition}\\
&y(0,x) = y_0(x).\\
&(\frac{\partial^2}{\partial t^2} - \Delta)y(t,x) = f(t,x) + u(t,x) \quad \text{on} \quad]0,T[\times\Omega, \text{ with initial conditions}\\
&y(0,x) = y_0(x) \;\; \frac{\partial}{\partial t}y(0,x) = y_1(x).
\end{aligned} \tag{2}$$

The spatial domain is Ω and the time domain $]0,T[$. We choose, for simplicity, a Dirichlet boundary condition on $\Gamma = \partial\Omega$:

$$\begin{aligned}
&y(x) = g(x) \;\; on\; \Gamma && \text{or}\\
&y(t,x) = g(t,x) \;\; on\;]0,T[\times\Gamma && \text{for the last two equations}
\end{aligned} \tag{3}$$

In the first case y is independent of time, U is taken as a convex subset of $L^2(\Omega)$ and

$$J(u,y) = \frac{1}{2}\int_{\Omega} |y(x)|^2 + \alpha|u(x)|^2 dx. \tag{4}$$

For the time-dependent problems we take U as a convex subset of $L^2(]0,T[\times\Omega)$ and

$$J(u,y) = \frac{1}{2}\int_{]0,T[\times\Omega} |y(x)|^2 + \alpha|u(t,x)|^2 dxdt. \tag{5}$$

These cost functions are a compromise between the desired damping of a physical quantity y and the cost of controlling the system represented by u term. A positive penalization parameter α controls this trade-off. The set of admissible controls U takes into account the possible constraints on the control which we restrict to be local in space and time and linear. For more on optimal control problems and also on mathematical issues such as well posedness, we refer to [Lio68] where can be found, in particular, the following reformulation of these problems as *optimality systems*. The solution is given by the resolution of (2)–(3), called the direct equations, together with (respectively)

$$\begin{aligned}
&-\Delta p(x) = y(x) \quad \text{on} \quad \Omega\\
&(-\frac{\partial}{\partial t} - \Delta)p(t,x) = y(t,x) \quad \text{on} \quad]0,T[\times\Omega, \text{ with initial condition}\\
&p(T,x) = 0.\\
&(\frac{\partial^2}{\partial t^2} - \Delta)p(t,x) = y(t,x) \quad \text{on} \quad]0,T[\times\Omega, \text{ with initial conditions}\\
&p(T,x) = 0 \;\; \frac{\partial}{\partial t}p(T,x) = 0.
\end{aligned} \tag{6}$$

$$\begin{aligned} &p(x) = 0 \quad \text{on} \quad \Gamma && \text{or} \\ &p(t,x) = 0 \quad \text{on} \quad]0,T[\times\Gamma && \text{for the last two equations} \end{aligned} \tag{7}$$

called the adjoint equations (backward in time) and the optimality conditions

$$\begin{aligned} &\int_\Omega (p + \alpha\, u)(v - u)\; dx \geq 0, \;\; \forall v \in U, && \text{or} \\ &\int_{]0,T[\times\Omega} (p + \alpha\, u)(v - u)\; dxdt \geq 0, \;\; \forall v \in U && \text{for the evolution problems.} \end{aligned} \tag{8}$$

From now on, we drop the dependence in space and time in the notation, these being implicitly defined by the type of problem we consider.

3 The Domain Decomposition Method

We decompose the domain as follows: Let Ω_i, $i = 1, m$ be a partition of Ω (i.e. $\Omega = \cup_i \bar{\Omega}_i$ and $\Omega_i \cap \Omega_j = \emptyset$ for $i \neq j$). We denote $\Gamma_i = \partial\Omega_i \cap \partial\Omega$ as the 'exterior' boundary of the subdomains and $\Sigma_{ij} = \partial\Omega_i \cap \partial\Omega_j$ as the interfaces. The external normal of $\partial\Omega_i$ is noted ν_i. We assume that the geometry of this decomposition is, as Ω, regular enough to ensure the well posedness of the global problems and the local subproblems defined by the DDM.

We schematically describe the *iterative* methods. At each step $n+1$ we perform local resolutions of (2), (3), (6), (7) and (8) where Ω has systematically been replaced by Ω_i in (2), (6) and (8) and Γ by Γ_i in (3) and (7). We call the local solutions at step $n+1$ on subdomain Ω_i: $(y_i^{n+1}, p_i^{n+1}, u_i^{n+1})$.

For instance, the local optimality conditions on Ω_i at step $n+1$ are obtained by replacing (8) with

$$\begin{aligned} &\int_{\Omega_i} (p_i^{n+1} + \alpha\, u_i^{n+1})(v_i - u_i^{n+1})\; dx \geq 0, \;\; \forall v_i \in U_i, && \text{or} \\ &\int_{]0,T[\times\Omega_i} (p_i^{n+1} + \alpha\, u_i^{n+1})(v - u_i^{n+1})\; dxdt \geq 0, \;\; \forall v_i \in U_i, && \end{aligned} \tag{9}$$

where U_i is a set of local admissible controls satisfying the same local constraints as the elements of U.

The only missing ingredients are the transmission conditions on the interfaces between subdomains (i.e. the new boundaries generated by the decomposition of the domain). They provide the mechanism which links the resolutions between neighboring subdomains at successive iteration steps. The choice of these two boundary conditions (one for the direct equation and one for the adjoint) is discussed in detail in the above mentioned references and are the same SRC transmissions conditions for all considered PDEs:

$$\begin{aligned} \frac{\partial}{\partial\nu_i} y_i^{n+1} + \beta\, p_i^{n+1} &= -\frac{\partial}{\partial\nu_j} y_j^n + \beta\, p_j^n \quad \textit{on } \Sigma_{ij}, \\ \frac{\partial}{\partial\nu_i} p_i^{n+1} - \beta\, y_i^{n+1} &= -\frac{\partial}{\partial\nu_j} p_j^n - \beta\, y_j^n \quad \textit{on } \Sigma_{ij}. \end{aligned} \tag{10}$$

The choice of β, a positive parameter, is discussed in section 6. We take $\beta = 1$ in the next section to simplify the presentation of a unified proof of convergence.

Let us finally mention that these local problems can be reformulated as optimal control problems and local cost functions can be derived depending on the original quantities to be minimized but also on a new "transmission" cost. This new cost arises because of the coupling in the transmissions conditions (10). This shows, as stated in the introduction, that we actually decompose the full optimization problem.

4 Convergence

The convergence is established using the equation on the local errors:

$$(\tilde{y}_i^{n+1}, \tilde{p}_i^{n+1}, \tilde{u}_i^{n+1}) = (y, p, u) - (y_i^{n+1}, p_i^{n+1}, u_i^{n+1}) \tag{11}$$

on each subdomain. These errors satisfy the same equations as $(y_i^{n+1}, p_i^{n+1}, u_i^{n+1})$ with $f = g = y_0 = y_1 = 0$. We *assume* (see above mentioned references for more details on mathematical issues) that the regularity of the global and local solutions allow the computations made in this section.

We introduce the following notations

$$\begin{aligned} \|z\|_{ij}^2 &= \int_{\Sigma_{ij}} z^2 \, d\sigma \ , \quad (z, z')_{ij} = \int_{\Sigma_{ij}} zz' \, d\sigma \\ \|z\|_i^2 &= \int_{\Omega_i} z^2 \, dx \ , \quad (z, z')_i = \int_{\Omega_i} zz' \, dx \end{aligned} \tag{12}$$

for the elliptic problem and

$$\begin{aligned} \|z\|_{ij}^2 &= \int_{]0,T[\times\Sigma_{ij}} z^2 \, d\sigma dt \ , \quad (z, z')_i = \int_{]0,T[\times\Sigma_{ij}} zz' \, d\sigma dt \\ \|z\|_i^2 &= \int_{]0,T[\times\Omega_i} z^2 \, dx dt \ , \quad (z, z')_i = \int_{]0,T[\times\Omega_i} zz' \, dx dt \end{aligned} \tag{13}$$

for the last two cases.

We are now able to give a general proof of convergence. To this end we introduce the following "energy" with support on the interfaces (note that each interface is counted twice, we have indeed $\Sigma_{ij} = \Sigma_{ji}$):

$$E^{n+1} = \sum_{i \neq j, \, s.t. \Sigma_{ij} \neq \emptyset} \{ \| \frac{\partial}{\partial \nu_i} \tilde{y}_i^{n+1} \|_{ij}^2 + \| \tilde{p}_i^{n+1} \|_{ij}^2 + \| \frac{\partial}{\partial \nu_i} \tilde{p}_i^{n+1} \|_{ij}^2 + \| \tilde{y}_i^{n+1} \|_{ij}^2 \}. \tag{14}$$

Using (10), the energies can be shown to satisfy (recall that we take $\beta = 1$)

$$\begin{aligned} E^{n+1} &= E^n - 2 \\ \sum_{i \neq j, \, s.t. \Sigma_{ij} \neq \emptyset} & \{ (\frac{\partial}{\partial \nu_i} \tilde{y}_i^{n+1}, \tilde{p}_i^{n+1})_{ij} - (\frac{\partial}{\partial \nu_i} \tilde{p}_i^{n+1}, \tilde{y}_i^{n+1})_{ij} \\ & + (\frac{\partial}{\partial \nu_i} \tilde{y}_i^n, \tilde{p}_i^n)_{ij} - (\frac{\partial}{\partial \nu_i} \tilde{p}_i^n, \tilde{y}_i^n)_{ij} \}. \end{aligned} \tag{15}$$

The last terms in the above expression are evaluated as follows. On each subdomain, the direct equation (in $\tilde{y}_i^{n+1}$) is multiplied by $\tilde{p}_i^{n+1}$ and the adjoint equation (in $\tilde{p}_i^{n+1}$)

is multiplied by $-\tilde{y}_i^{n+1}$. We then integrate by part in space and integrate in space and time in the case of evolution equations. Adding the results, we note that the terms involving time derivatives and gradients vanish and finally obtain, for all i (regardless of the considered PDE):

$$\begin{aligned} -\sum_{j,s.t.\,\Sigma_{ij}\neq\emptyset} \{(\frac{\partial}{\partial\nu_i}\tilde{y}_i^{n+1},\tilde{p}_i^{n+1})_{ij} - (\frac{\partial}{\partial\nu_i}\tilde{p}_i^{n+1},\tilde{y}_i^{n+1})_{ij}\} = \\ -\|\tilde{y}_i^{n+1}\|_i^2 + (\tilde{p}_i^{n+1},\tilde{u}_i^{n+1})_i \end{aligned} \tag{16}$$

We now use the optimality conditions. We choose $v = u_i^{n+1}$ on Ω_i and 0 elsewhere in (8) and $v_i = u$ in (9) and subtract the two inequalities. This yields the estimate

$$(\tilde{p}_i^{n+1},\tilde{u}_i^{n+1})_i \leq -\alpha\|u_i^{\tilde{n}+1}\|_i^2. \tag{17}$$

Combining (16) and (17) (and similar results for step n) and (15), we establish a law of decrease for the energy:

$$E^{n+1} \leq E^n - 2\sum_i\{\|\tilde{y}_i^{n+1}\|_i^2 + \alpha\|\tilde{u}_i^{n+1}\|_i^2 + \|\tilde{y}_i^n\|_i^2 + \alpha\|\tilde{u}_i^n\|_i^2\}. \tag{18}$$

Summing over n gives straightforwardly a first result of convergence on the errors on each subdomain Ω_i:

$$\|\tilde{y}_i^{n+1}\|_i \xrightarrow{n} 0 \quad \|\tilde{u}_i^{n+1}\|_i \xrightarrow{n} 0. \tag{19}$$

This result can be improved using the equations and the uniform boundedness of E^n.

5 Other Problems

The convergence of this method relies on the SRC transmission conditions and the structure of optimal control problems reformulated as coupled system formed of a direct and adjoint equations and a optimality condition. The *convexity* of the cost function provides the necessary coercivity for this system. This explains why this method can be applied to a wide range of linear optimal control problems involving more complicated (possibly nonsymmetric and inhomogeneous) operators for the equation and the boundary conditions. Boundary observation and control problems can also be treated. Dealing with nonlocal observation is also possible. It however couples the resolution of subproblems set on the domain of observation.

This DDM can be applied to noncoercive PDEs such as the Helmholtz equation. The proof of convergence remains formally the same but the usual bilinear form is replaced by a sesquilinear form.

We are currently working on the adaptation of this method to the decomposition of the HUM method for exact controllability problems [Lio88].

Let us mention that it is also possible, at least formally, to use the same SRC transmision conditions to domain decompose in time.

6 Remarks

Numerical Implementation and Speed of Convergence

A good way of discretizing these problem is to use mixed hybrid finite elements (see [CJ86] or [RG90a, RG88] on the use of mixed finite elements in domain decomposition methods). This approach is well suited to our problem for it uses in particular, as degrees of freedom, the fluxes of the normal derivatives and the average values of the trace of the direct and adjoint states on the interfaces which are the natural unknowns of our transmission conditions. The proof of convergence in the continuous case is easily extended to this mixed hybrid discrete formulation as it allows an exact discrete integration by part.

The parameter β has a decisive influence on the speed of convergence. We always choose it proportional to $\frac{1}{h}$ where h is the size of the finite elements. The discrete transmission conditions are in these case adimensional.

It was shown in [Des93] (for the direct Helmholtz equation) that the eigenvalues of the discrete iteration operator may be close to 1. This explains the observed bad convergence behavior of the algorithm. A simple way to remedy to this situation (still [Des93]) is to use and under-relaxed version of the transmission conditions. For control problems they take the form

$$\begin{aligned} \frac{\partial}{\partial\nu_i} y_i^{n+1} + \beta\, p_i^{n+1} &= \gamma(-\frac{\partial}{\partial\nu_j} y_j^n + \beta\, p_j^n) + (1-\gamma)(\frac{\partial}{\partial\nu_i} y_i^n + \beta\, p_i^n) \quad \mathit{on}\ \Sigma_{ij}, \\ \frac{\partial}{\partial\nu_i} p_i^{n+1} - \beta\, y_i^{n+1} &= \gamma(-\frac{\partial}{\partial\nu_j} p_j^n - \beta\, y_j^n) + (1-\gamma)(\frac{\partial}{\partial\nu_i} p_i^n - \beta\, y_i^n) \quad \mathit{on}\ \Sigma_{ij} \end{aligned} \tag{20}$$

The relaxation parameter γ has to belong to $]0,1[$ (we usually choose $\gamma = \frac{1}{2}$). The theoretical convergence proof can also be established with (20) instead of (10).

More pragmatically, the convergence speed observed in actual simulations is always proportional to the number of subdomains. Multilevel or multigrid method are much faster for standard direct elliptic problems (see [SBG96] for instance). This is not so evident for hyperbolic problems or noncoercive elliptic equations such as the Helmholtz equation without even speaking of extending this methods to the resolution of control problems.

Resolution of the Subproblems

Our own approach on the decomposition of the domain and the resolution of the subproblems was to take the smallest possible subdomains, i.e. each finite element is a subdomain. This is of course the worst possible choice in terms of number of iterations but it restricts the number of degrees of freedom on each subdomain to a minimum and allows in most cases an analytical resolution of the subproblems. This strategy is well suited to an implementation on a massively parallel machine but requires a structured meshing of the domain.

In the case of complicated geometries, a domain decomposition in simple shapes may be a good motivation to use this domain decomposition method. A method of resolution of the subproblems is still needed.

As already mentioned in section 2, in the case of evolution equations the adjoint equation is backward in time. Gradient-type methods rely on iterating successively forward integrations for the direct equation and backward integrations for the adjoint equation. The adjoint variable providing a descent direction for the optimization method. An other possibility is to compute the *feed-back law* which express the linear relationship between p and y. It can *formally* be written $\{p(t), \frac{\partial}{\partial t}p(t)\} = A(t) * \{y(t), \frac{\partial}{\partial t}y(t)\} + \{r0(t), r1(t)\}$ [Lio68]. At each instant t, an operator $A(t)$ maps a space-dependent function into an other space-dependent function. If A and r can be computed, the feed-back law can be used to eliminate p in the direct equation. The operators $A(t)$ satisfy a nonlinear Riccati differential equation which can be difficult to solve at least because of the size of the discretization of $A(t)$. The resolution of a PDE depending on the second hand terms of the original system (and on A) gives r. A similar feed-back law $\{p_i^{n+1}(t), \frac{\partial}{\partial t}p_i^{n+1}(t)\} = A_i^{n+1}(t) * \{y_i^{n+1}(t), \frac{\partial}{\partial t}y_i^{n+1}(t)\} + \{r0_i^{n+1}(t), r1_i^{n+1}(t)\}$ can also be defined for each of the subproblems of our DDM. The important remark here is that A_i^{n+1} *does not depend on the iterative process* but only on the geometry of the decomposition and only need to be computed once. This is because it defines the homogeneous part of the feed back law. The DDM therefore implicitly decompose the computation of the feed-back law. A numerical implementation of the method using this technique is presented in [Ben97].

REFERENCES

[AB73] A. Bensoussan R. Glowinsky J. L. (1973) Méthode de décomposition appliquée au contrôle optimal de systèmes distribués. In *5th IFIP Conference on Optimization techniques.* Lecture Notes in Computer Science.

[BD96] Benamou J. and Després B. (1996) A domain decomposition method for the Helmholtz equation and related optimal control problems. *Submitted to J. Comp. Physics and INRIA Tech. Report 2791* .

[Ben93] Benamou J. (1993) Décomposition de domaine pour le contrôle optimal de systèmes gouvernés par des équations aux dérivées partielles elliptiques. *C. R .Acad. Sci. Paris* 317: 205–209.

[Ben95a] Benamou J. (1995) A domain decomposition method for the optimal control of system governed by the helmholtz equation. In Cohen G. (ed) *Third international conference on mathematical and numerical wave propagation phenomena (Juan-les-Pins.* SIAM.

[Ben95b] Benamou J. (1995) A massively parallel algorithm for the optimal control of systems governed by elliptic p.d.e.'s. In *seventh SIAM conference on parallel processing for scientific computing.* SIAM.

[Ben96a] Benamou J. (1996) Décomposition de domaine pour le contrôle de systèmes gouvernés par des équations d'évolution. *Submitted to C. R .Acad. Sci. Paris* .

[Ben96b] Benamou J. (1996) Domain decomposition methods with coupled transmission conditions for the optimal control of systems governed by elliptic partial differential equations. *to appear in SINUM and INRIA Tech. Report 2246* .

[Ben97] Benamou J.-D. (1997) Optimal control of systems governed by the wave equation : resolution of a test case using a domain decomposition method. Technical report, INRIA.

[Bou] Bounaim A.In *This proceedings.*

[CR91] Chavent G. and Roberts J. (December 1991) A unified physical presentation of mixed, mixed-hybrid finite elements and standard finite difference approximations for the determination of velocities in waterflow problems. *Advances in Water Ressources* 14(6): 329–348.

[Des91] Després B. (1991) Domain decomposition method and the Helmholtz problem. In Cohen G., Halpern L., and Joly P. (eds) *Mathematical and Numerical Aspects of*

Wave Propagation Phenomena, pages 44–52. SIAM.

[Des93] Després B. (1993) Domain decomposition method and the Helmholtz problem (part ii). In Kleinman R., Angell T., Colton D., Santosa F., and Stackgold I. (eds) *Second international conference on mathematical and numerical aspects of wave propagation phenoma.* SIAM.

[Leu96] Leugering G. (1996) On dynamic domain decomposition of controlled networks of strings and joint-masses.

[Lio68] Lions J. (1968) *Contrôle optimal de systèmes gouvernés par des équations aux dérivées partielles.* Dunod, Paris.

[Lio88] Lions J. (1988) *Contrôlabilité exacte perturbations et stabilisation de systèmes distribués, tome 1.* Masson, Paris.

[Lio90] Lions P. (1990) On the Schwarz alternating method 3. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Third international symposium on domain decomposition methods for partial differential equations.* SIAM.

[RG88] R. Glowinski M. W. (1988) Domain decomposition and mixed finite element methods for elliptic problems. In Glowinski R., Golub G. H., Meurant G. A., and Périaux J. (eds) *First international symposium on domain decomposition methods for partial differential equations.* SIAM.

[RG90a] R. Glowinski M.Kinton M. W. (1990) Acceleration of domain decomposition algorithms for mixed finite element methods by multi level methods. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Third international symposium on domain decomposition methods for partial differential equations.* SIAM.

[RG90b] R. Glowinski P. L. (1990) Augmented lagrangian interpretation of the nonoverlapping Schwarz alternating method. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *third international symposium on domain decomposition methods for partial differential equations.* SIAM.

[SBG96] Smith B., Bjorstad P., and Gropp W. (1996) *Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press.

[W.H79] W.Hackbush (1979) On the fast solving of parabolic boundary control problems. *SIAM J. Control and Optimization* 17: 231–244.

32

Spectral Elements on Infinite Domains

Kelly Black

1 Introduction

The use of infinite computational domains for the approximation of partial differential equations (PDEs) can arise in many applications. For example, in the approximation of external flows, or the approximation of some geophysical flows, the flow field may be extended to an infinite domain to simplify the treatment of boundaries. Also, electromagnetic fields may have an influence over an extremely large domain, or because of the relatively strong forces on small intervals they may be best approximated by extending the domain to an infinite interval [Bel94, Lan60, Sec95, Wil93, Wil94]. This may be particularly true when the treatment of boundary conditions can lead to artificial reflections that may pollute the approximation near the origin.

Two methods to approximate PDEs on infinite domains are examined here. Both methods are spectral element techniques and rely on high-order polynomials to construct the local approximation within the individual subdomains. To compare the two methods, a simple one-dimensional Helmholtz equation is examined:

$$\begin{aligned} u_{xx} + \lambda u &= f(x), \qquad -\infty < x < \infty \qquad (1) \\ \lim_{x \to \infty} u(\pm x) &= 0. \end{aligned}$$

For both methods, the computational domain is divided into non-overlapping subdomains. Near the origin, finite subdomains are employed, while far from the origin, semi-infinite subdomains are employed. Both methods rely on local spectral approximations and are not collocation methods.

The first method examined is based on a Laguerre polynomial expansion while the second method is based on a mapping of the semi-infinite interval to a finite interval proposed by Boyd [Boy87] for a single domain approximation. The local approximation is found as a linear combination of basis elements that are constructed from the Legendre polynomials. These basis functions are based on those proposed by Shen [She94]. Unlike the method proposed by Karageorghis and Phillips [Kar87],

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

a variational method is constructed in which the basis functions are divided between polynomials which are zero on the boundaries and polynomials which are not zero on the boundary.

In this presentation, the basis functions are defined first. Once done, two methods for extending the method to accommodate semi-infinite domains are given. The first employs Laguerre polynomials while the second employs the use of the mapping method proposed by Boyd [Boy87]. A direct comparison of the two methods is given. In the comparisons, equations are examined in which the true solution decays to zero exponentially and a true solution that decays only algebraicly.

2 Introduction to Spectral Elements with Local Spectral Basis

To introduce the method, a simple example is given. A one dimensional Helmholtz equation is examined on a finite domain:

$$\begin{aligned} u_{xx} + \lambda u &= f(x), \qquad -1 < x < 1 \\ u(\pm 1) &= 0. \end{aligned} \tag{2}$$

To construct an approximation for this example, the domain, $-1 \leq x \leq 1$, is divided into two subdomains, [-1,0] and [0,1].

The global approximation is found from the space of piecewise polynomials. For example, in a given subdomain the local approximation is a polynomial up to a given degree, N. This approximation is found as a linear combination of polynomials which are divided into those that are zero on the boundary and those that are not. The choice for these test functions is motivated by the results of Shen [She94]:

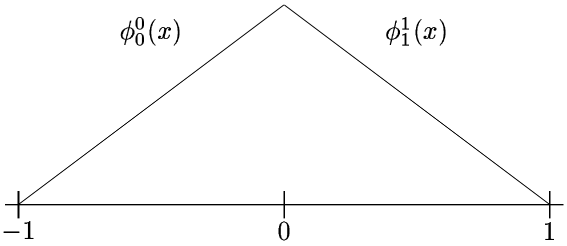
$$\phi_i(x) = \begin{cases} \frac{1+x}{2} & i = 0, \\ \frac{1-x}{2} & i = 1, \\ L_i(x) - L_{i-2}(x) & i > 1, \end{cases} \tag{3}$$

for $i = 0 \ldots N$, and $L_i(x)$ is the Legendre polynomial of degree i. The span of the test functions is the span of polynomials up to degree N, and each of the ϕ_i's are linearly independent. The primary difference from the work of Shen [She94] is the addition of ϕ_0 and ϕ_1 and implementing it as a spectral element method. The two new linear basis functions are used to form a hat function whose support includes adjacent subdomains (see Figure 1).

The linear basis functions, $\phi_0(x)$ and $\phi_1(x)$, allow for a straightforward method to handle boundary conditions. Unlike collocation methods such as the one proposed by Karageorghis and Phillips [Kar87], the method is closer in spirit to the methods originally proposed by Patera [Pat84], and the discretization requires that only C^0 continuity at the subdomain interfaces be enforced.

The choice of basis functions also yield a symmetric tridiagonal mass matrix and a diagonal stiffness matrix. As pointed out by Shen [She94], a linear combination of the

Figure 1 Example of two subdomains, [-1,0] and [0,1], on the interval [-1,1]. Trial functions $\phi_0^0(x)$ and $\phi_1^1(x)$ combine on adjacent subdomains to assemble a "hat" function on adjacent subdomains.



Legendre polynomials satisfies the following identity:

$$\begin{aligned} -\int_{-1}^{1} \left(L_i(x) - L_{i-2}(x)\right)' \left(L_i(x) - L_{i-2}(x)\right)' dx \qquad\qquad (4) \\ = -\int_{-1}^{1} (2i-1)\, L_{i-1}(x)\, (2j-1)\, L_{j-1}(x) dx, \\ = -\,(2i-1)\,(2j-1) \int_{-1}^{1} L_{i-1}(x) L_{j-1}(x) dx, \\ = -2\,(2i-1)\,\delta_{ij}. \end{aligned}$$

The final equality follows from the orthogonality of the Legendre polynomials. The result is a nearly diagonal stiffness matrix for the one dimensional problem (see Figure 2). The mass matrix is tridiagonal, and since a tensor product is used for higher dimensions, the result is a sparse linear system.

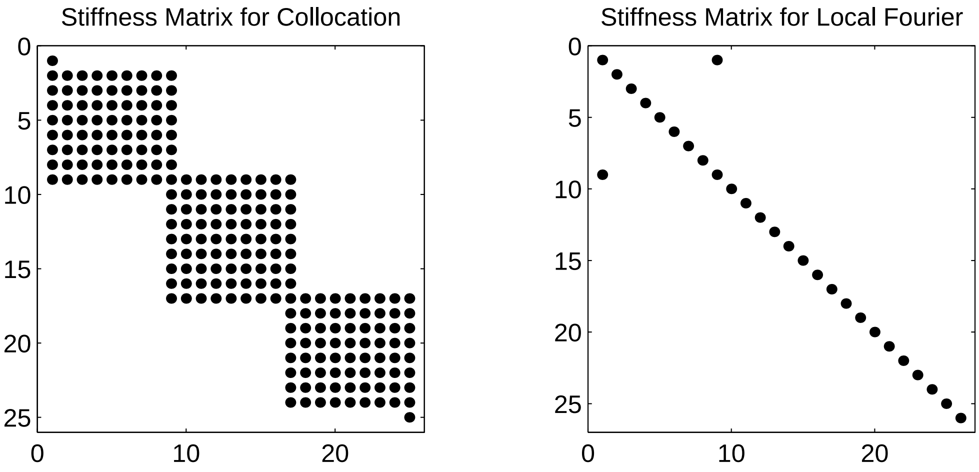
3 Laguerre Polynomials

The Laguerre polynomials are orthogonal on a semi-infinite interval, and the family of polynomials represents a natural candidate to construct an approximation on a semi-infinite domain. The Laguerre polynomials, denoted $L_i^{(0)}(x)$ for the Laguerre polynomial of degree i, are orthogonal on the semi-infinite domain with respect to the weight function e^{-x} [Fun92]:

$$\int_0^{\infty} L_i^{(0)}(x) L_j^{(0)}(x) e^{-x} dx \;=\; \delta_{ij}. \qquad (5)$$

The orthogonality relationship has made the Laguerre polynomials the focus of a great deal of attention. In particular, both Funaro [Cou90] and Maday [Mad85] have shown many theoretical results on the accuracy of methods utilizing Laguerre polynomials. However, in practice there are also many difficulties associated with Laguerre polynomials [Mav89]. First, the Laguerre polynomials scale badly for the larger degree polynomials [Fun89]. Moreover, the Laguerre polynomials suffer in the

Figure 2 Comparison of the stiffness matrix for the collocation and the local spectral schemes.



small range of boundaries that can be accommodated at infinity [Mav89]. Finally, the Laguerre polynomials experience spectral convergence only for solutions that decay to zero exponentially [Boy87, Kar87].

Despite these drawbacks, Laguerre polynomials offer a simple and elegant implementation for the approximation of solutions which decay to zero fast enough [Mad85]. For example, for a spectral element method, semi-infinite domains can be accommodated through the following basis functions:

$$\phi_0(x) = L_0^{(0)}(x)e^{-x/2}, \tag{6}$$

$$\phi_i(x) = \left(L_n^{(0)}(x) - L_{n-1}^{(0)}(x)\right) e^{-x/2}, \quad i > 0. \tag{7}$$

A variational approximation can be constructed resulting in symmetric, tridiagonal mass and stiffness matrices. This process is made easier since, as defined, the basis functions satisfy the following identities,

$$\phi_0(0) = 1, \tag{8}$$

$$\phi_i(0) = 0, \qquad i > 0, \tag{9}$$

$$L_0^{(0)}(x) = 1, \tag{10}$$

$$\frac{d}{dx}\left(L_i^{(0)}(x) - L_{i-1}^{(0)}(x)\right) = -L_{i-1}^{(0)}(x), \qquad i > 0. \tag{11}$$

The basis functions satisfy similar boundary conditions as the the basis functions for the finite subdomains. Continuity at the interface is handled through the first basis function, $\phi^0(x)$, in the same manner as for the finite subdomains.

4 Mapping to a Finite Subdomain

Another common method for building an approximation on a semi-infinite interval is to make use of an algebraic mapping. By mapping the semi-infinite interval, $[0, \infty)$, to a finite interval, $[-1, 1]$. Orthogonal polynomial methods can be utilized to construct an approximation on the resulting finite interval.

We propose to adapt the mapping proposed by Boyd [Boy87]:

$$\begin{aligned} y &= M\frac{1+x}{1-x}, & (12)\\ x &= \frac{y-M}{y+M}. & (13) \end{aligned}$$

While the method has been employed for the single domain case, we propose to extend its use to the multidomain approach in a manner similar to that proposed by Karageorghis and Phillips [Kar87].

For example, given a simple 1D Helmholtz equation, the area near the origin can be approximated through the use of finite subdomains. Away from the origin, semi-infinite subdomains can be employed. For the semi-infinite subdomains, the following functions are defined (the notation introduced by Boyd [Boy87] is employed here),

$$\begin{aligned} LM_n(y) &= L_n\left(\frac{y-M}{y+M}\right), & (14)\\ &= L_n(x), \end{aligned}$$

where $L_n(x)$ is the n^{th} Legendre polynomial. To take advantage of the orthogonality of the Legendre polynomials, a weight function is required,

$$\begin{aligned} \int_0^\infty LM_n(y)LM_m(y)\frac{2M^2}{(y+M)^2}dy &= \frac{M}{2}\int_{-1}^1 L_n(x)L_m(x)dx, & (15)\\ &= \frac{M}{2n+1}\delta_{nm}. \end{aligned}$$

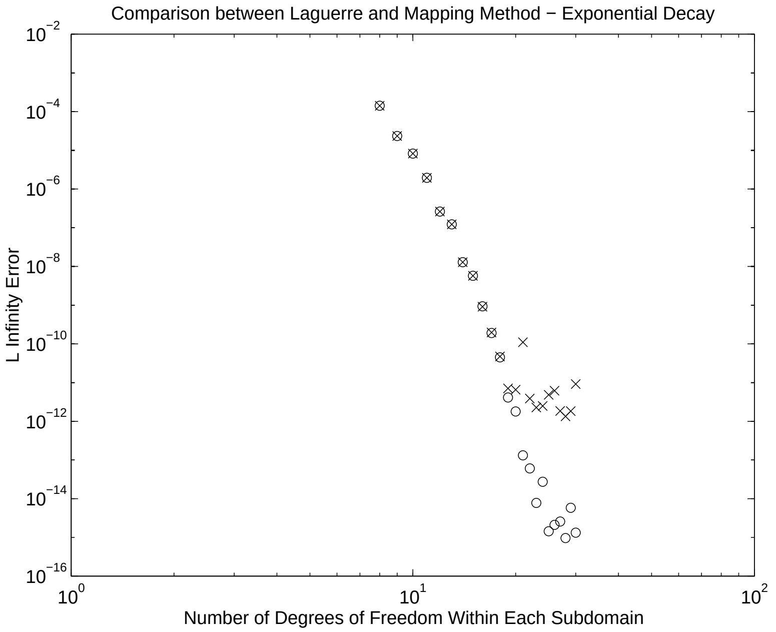
The test functions are chosen to remain consistent with those found in Section 2:

$$\phi M_j(y) = \begin{cases} \frac{1+x}{2} & j=0,\\ \frac{1-x}{2} & j=1,\\ L_j(x) - L_{j-2}(x) & j>1. \end{cases} \qquad (16)$$

The continuity conditions remain identical to those found for the finite subdomain, and the choice of basis functions leads to a mass matrix that is identical to that found for a finite subdomain. The stiffness matrix is not as elegant, though. The stiffness matrix, while not diagonal, is a banded matrix, with band width 5. While the new stiffness matrix is not symmetric, the new approximation can be employed for a much wider collection of boundary conditions at infinity. In fact, the boundaries are enforced in exactly the same way as is done for a finite domain.

The stiffness matrix for the semi-infinite subdomain is found in the same manner

Figure 3 Spectral element approximation of a solution with exponential decay using Laguerre polynomials and algebraic mappings on the outer subdomains. The errors for the Laguerre polynomials are denoted by ×, while the errors for the mapping to the finite interval are denoted by ∘.

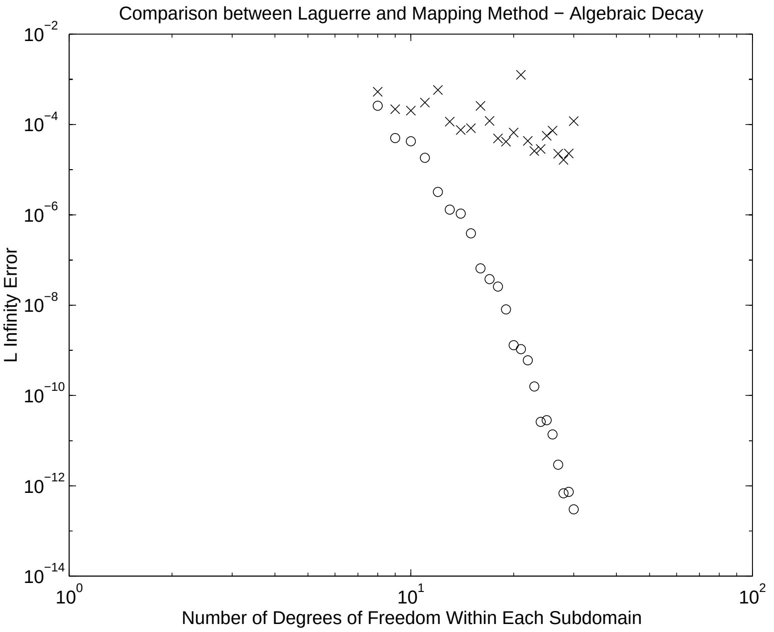


as that for a finite subdomain,

$$
\begin{aligned}
\left(\mathcal{S}_N\right)_{mj} &= -\int_0^\infty \frac{d}{dy}\left(\phi M_j(y)\right)\frac{d}{dy}\left(\phi M_m(y)\frac{2M^2}{(y+M)^2}\right)dy, \qquad (17)\\
&= -\frac{M}{2}\int_{-1}^{1}\phi_j'(x)\frac{(1-x)^2}{2M}\phi_m'(x)\frac{(1-x)^2}{2M}dx,\\
&\quad +\frac{M}{2}\int_{-1}^{1}\phi_j'(x)\phi_m(x)\frac{(1-x)^3}{2M^2}dx.
\end{aligned}
$$

Note that the weight function is similar to that proposed by Boyd [Boy87] but differs slightly. The weight function includes the square of the mapping parameter in the numerator to insure that the weight function is continuous across the subdomain interface. This choice for the weight function insures that only C^0 continuity need be enforced across the subdomain interface in the resulting variational formulation.

Figure 4 Spectral element approximation of a solution with algebraic decay using Laguerre polynomials and algebraic mappings on the outer subdomains. The errors for the Laguerre polynomials are denoted by ×, while the errors for the mapping to the finite interval are denoted by ∘.



Comparison of the Two Methods

A comparison of the two methods for two situations is examined. In both of these examples, a 1D Helmholtz equation (Equation 1) is examined (with zero boundaries), and in both cases $\lambda = 2$. In the first example the true solution is e^{-x^2}, and in the second example the true solution is $\frac{1}{1+x^2}$.

The interval $[-5, 5]$ is divided into four equal finite subdomains, and the remaining two subdomains are $(-\infty, 5]$ and $[5, \infty)$. In this situation, the basis functions on subdomain n, $\phi_i^n(x)$, only have support on subdomain n for $i = 2 \ldots N$. The functions $\phi_0^n(x)$ and $\phi_1^n(x)$ are used to construct basis functions whose support only includes adjacent subdomains.

The L^∞ errors for the two trials are shown in Figures 3 and 4. The approximation that utilizes Laguerre polynomials does exhibit fast convergence for the equation whose solution decays to zero exponentially (Figure 3). However, this is not the case for the approximation to the equation whose solution does not decay as fast (Figure 4). The method using the proposed mapping, though, does exhibit similar convergence properties for both situations.

To test the mapping method in a two dimensional case, a Poisson equation is

$N_x = N_y$	L^∞ Error
8	5.206259e-04
10	2.438615e-04
12	1.365491e-04
14	7.567740e-05
16	4.658750e-05

Table 1 Maximum Errors for the approximation of a Poisson equation on an infinite domain. For each approximation 16 subdomains are employed, and the polynomial degree is given for each trial. For each trial the polynomial degree for both the x and y direction are equal.

examined,

$$\begin{aligned} \triangle u &= \frac{4x^2+4y^2-4}{\left(1+x^2+y^2\right)^3}, \\ \lim_{x\to\infty} u(\pm x, y) &= 0, \\ \lim_{y\to\infty} u(x, \pm y) &= 0. \end{aligned} \tag{18}$$

The solution to this equation, $\frac{1}{1+x^2+y^2}$, decays to zero algebraically. In this test case 16 subdomains are employed. Four finite subdomains are employed on the unit square, $[-1,1]\times[-1,1]$. Away from the unit square semi-infinite subdomains are employed to construct an approximation. The errors are shown in Table 1, and the errors are the maximum errors found on the abscissa of the Legendre-Gauss quadrature within each subdomain.

REFERENCES

[Bel94] Bellan, P.M. (November 1994) Alfen 'Resonance' Reconsidered: Exact equations for wave propogation across a cold inhomogeneous plasma. *Physics of Plasmas* 1(11): 3523–3541.

[Boy87] Boyd, John (1987) Orthogonal Rational Functions on a Semi-Infinite Interval. *Journal of Computational Physics* 70: 63–68.

[Cou90] Coulaud, O., D. Funaro, and O. Kavian (1990) Laguerre Spectral Approximations of Elliptic Problems in Exterior Domains. *Computer Methods in Applied Mechanics and Engineering* 80(1-3): 451–458.

[Fun89] Funaro, D. (October 1989) Computational Aspects of Pseudospectral Laguerre Approximations. Technical Report NAS1-18605, ICASE, NASA Langley Research Center, Hampton VA 23665-5225.

[Fun92] Funaro, D. (1992) *Polynomial Approximation of Differential Equations.* Springer-Verlag, Berlin.

[Kar87] Karageorghis, A., and T.N. Phillips (1987) Spectral Collocation Methods for Stokes Flow in Contraction Geometries and Unbounded Domains. *Journal of Computational Physics* 80: 314–330.

[Lan60] Landau, L. D., and E. M. Lifshitz (1960) *Electrodynamics of Continuous Media*, volume 8 of *Course of Theoretical Physics*. Pergamon Press, New York.

[Mad85] Maday, Y., B. Pernaud-Thomas, and H. Vandeven (1985) Reappraisal of Laguerre Type Spectral Methods. *La Recherche Aerospatial (English Addition)* 6: 13–35.

[Mav89] Mavriplis, C. (1989) Laguerre Polynomials for Infinite-Domain Spectral Elements. *Journal of Computational Physics* 80: 480–488.

[Pat84] Patera, Anthony T. (1984) A Spectral Element Method for Fluid Dynamics: Laminar Flow in a Channel Expansion. *Journal of Computational Physics* 54: 468–488.

[Sec95] Secchi, Paolo (1995) Well-Posedness for a mixed problem for the equations of ideal Magneto-Hydrodynamics. *Archivel der Mathematik* 64: 237–245.

[She94] Shen, J. (November 1994) Efficient Spectral-Galerkin Method I. Direct Solvers of Second and Fourth-Order Equations Using Legendre Polynomials. *SIAM Journal of Scientific Computing* 15(6): 1489–1505.

[Wil93] Wilson, S.K. (1993) The Effect of a Uniform Magnetic Field on the Onset of Marangoni Convection in a Layer of Conducting Fluid. *Quarterly Journal of Mechanics and Applied Mathematics* 46(2): 211–248.

[Wil94] Wilson, S.K. (November 1994) The effect of a uniform magnetic field on the onset of steady Marangoni convection in a layer of conducting fluid with a prescribed heat flux on its lower boundary. *Physics of Fluids* 6(11): 3591–3600.

33

A Lagrangian Approach to a DDM for an Optimal Control Problem

Aïcha Bounaïm

1 Introduction

We give a Lagrangian interpretation to a domain decomposition method for an optimal control problem governed by an elliptic partial differential equation. We minimize the cost function and the links between subdomains, the constraints at the interfaces being treated by augmented Lagrangian techniques. An algorithm is proposed and illustrated with numerical computations.

We consider the following distributed optimal control problem governed by an elliptic partial differential equation:

$$J(u) = \inf_{v \in U_{ad}} J(v), \; u \in U_{ad}, \tag{1}$$

where

$$J(v) = \frac{1}{2}(\int_\Omega (y(v) - y_d)^2 dx + \nu \int_\Omega v^2 dx),$$

with Ω is a bounded smooth open subset of $\mathbb{R}^q$, y_d is the desired state given in $L^2(\Omega)$, U_{ad} is a closed convex subset of $L^2(\Omega)$ (space of admissible controls), ν is a strictly positive real number, $y(v)$ is the solution of the system:

$$\left\{ \begin{array}{rcl} -\Delta y(v) & = & f + v \;\text{ in }\; \Omega, \\ y(v) & = & 0 \;\text{ on }\; \Gamma \;=\; \partial\Omega, \end{array} \right. \tag{2}$$

and $f \in L^2(\Omega)$.

Proposition 1 *The problem* (1) *has a unique solution u and the mapping $u \mapsto y(u)$ is affine continuous from U_{ad} into $H^1(\Omega)$.*

Proof. See [Lio68].

Remark 1 *In the unconstrained case, the control is given by $u = -\frac{p}{\nu}$ where $p = p(u)$ is the adjoint state, solution of:*

$$\left\{ \begin{array}{rcll} -\Delta p(u) & = & y(u) - y_d \; in & \Omega, \\ p(u) & = & 0 \; on & \Gamma \;=\; \partial\Omega. \end{array} \right. \tag{3}$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

In the general case, the problem (1) *is characterized by the following system called "optimality system":*

- *Direct state* (1).
- *Adjoint state* (3).
- *Optimality condition:* $\int_\Omega (p(u) + \nu u)(v - u)dx \geq 0 \qquad \forall v \in U_{ad}$.

It is also proved in [Lio68] that there exists a unique solution (u, y, p) *of the above optimality system and* u *is the minimizer of* (1).

2 Domain Decomposition for the Optimal Control Problem

We will be interested in a non-overlapping domain decomposition method. Thus, Ω is decomposed into m subdomains and we introduce the following notations:

$$\begin{array}{ll} \Omega = \bigcup_{i=1}^m \Omega_i \cup \Sigma, & \Gamma_i = \partial\Omega \cap \partial\Omega_i, \\ \sigma_{ij} = \partial\Omega_i \cap \partial\Omega_j, & \Sigma = \bigcup_{1 \leq i \neq j \leq m} \sigma_{ij}, \\ V_i = \{\ y_i \in H^1(\Omega_i);\ y_i = 0 \ in \ \Gamma_i\ \}, & \end{array}$$

where Ω_i are disjoint open sets in $\mathbb{R}^q$.

The idea of the domain decomposition method proposed is to define the augmented Lagrangian associated with the decomposed optimal control problem. Therefore, the continuity of function value at the interfaces is treated by Lagrangian techniques, whereas the flux continuity is formulated explicitly in the direct *pde* on each subdomain Ω_i:

$$\left\{ \begin{array}{rcll} -\Delta y_i & = & f_i + v_i & \text{in} \quad \Omega_i, \\ y_i & = & 0 & \text{on} \quad \Gamma_i, \\ \frac{\partial y_i}{\partial \eta} & = & \omega_{ij} & \text{on} \quad \sigma_{ij}, j \neq i. \end{array} \right. \tag{4}$$

where $\eta = \eta_{ij} = -\eta_{ji}$ is the unit outward normal to Ω_i on σ_{ij}, for $i < j$ and ω_{ij} is an extra variable introduced to ensure the continuity of the normal derivative and satisfies $\omega_{ij} = \omega_{ji} \in H^{-\frac{1}{2}}(\sigma_{ij})$ for $j \neq i$ and $\sigma_{ij} \neq \emptyset$.

Then, the cost functional has the new expression:

$$J(v_i, y_i) = \sum_{i=1}^m \frac{1}{2} \left[\int_{\Omega_i} (y_i - y_d^i)^2 dx + \nu \int_{\Omega_i} v_i^2 dx \right].$$

Finally, we obtain a new optimization problem [1]:

$$\left\{ \begin{array}{l} \text{Minimize} \quad J(v_i, y_i) \\ v_i \in U_{ad}^i \\ y_i \ \text{ and } \ v_i \ \text{ linked by (4)} \\ y_i = y_j \ \text{ on } \ \sigma_{ij}, \ j \neq i. \end{array} \right. \tag{5}$$

[1] We use the notation $(v_i, y_i) = ((v_i)_{1 \leq i \leq m}, (y_i)_{1 \leq i \leq m})$.

Lagrangian Formulation

Now, we introduce the augmented Lagrangian associated with the above minimization problem. In order to enforce and to decouple the constraint of the interface continuity, we add an extra-variable q_{ij} such that: $q_{ij} = q_{ji} \in H^{\frac{1}{2}}(\sigma_{ij})$, for $j \neq i$, $\sigma_{ij} \neq \emptyset$ and

$$y_i = y_j = q_{ij} \ on \ \sigma_{ij} \tag{6}$$

(see [GL90] for such consideration).
Then, after making explicit the constraint between v_i and y_i, the augmented Lagrangian L_r is given by [2]:

$$\begin{aligned} L_r(v_i, y_i, p_i, \omega_{ij}, \lambda_{ij}, q_{ij}) = J(v_i, y_i) - \sum_{i=1}^{m} \Big(\int_{\Omega_i} \nabla y_i \nabla p_i dx - \int_{\Omega_i} (f_i + v_i) p_i dx \Big) \\ + \sum_{1 \leq i < j \leq m} \int_{\sigma_{ij}} \omega_{ij}(p_i - p_j) d\sigma_{ij} + \sum_{i=1}^{m} \sum_{j,\ \sigma_{ij} \neq \emptyset} \int_{\sigma_{ij}} \lambda_{ij}(y_i - q_{ij}) d\sigma_{ij} \\ + \frac{r}{2} \sum_{i=1}^{m} \sum_{j,\ \sigma_{ij} \neq \emptyset} \int_{\sigma_{ij}} (y_i - q_{ij})^2 d\sigma_{ij}, \end{aligned}$$

where λ_{ij} are the Lagrange multipliers corresponding to the constraint (6) and r (augmented Lagrangian constant) is a positive real number.

Remark 2 *In the L_r expression, the ω_{ij} can be interpreted as Lagrange multipliers corresponding to the continuity of the adjoint state on the interface.*

Proposition 2 *If $(u_i, y_i, p_i, \omega_{ij}, \lambda_{ij}, q_{ij})$ is a saddle point of L_r, when it exists, then,*

$$u_i = u/\Omega_i, \ y_i = y/\Omega_i, \ p_i = p/\Omega_i. \tag{7}$$

where (u, y, p) is the saddle point of the Lagrangian associated with the problem (1) and u is the minimizer of (1).

Proof. In the proof, we explicit characterization of a saddle point of L_r; we denote: $(s.p) = (v_i, y_i, p_i, \omega_{ij}, \lambda_{ij}, q_{ij})$. As L_r is differentiable in each variable, for $i : 1 \leq i \leq m$, j such that $j \neq i$ and $\sigma_{ij} \neq \emptyset$, we have:

$$\begin{aligned} (\frac{\partial L_r}{\partial p_i}(s.p), \phi_i) &= 0 \quad \forall \phi_i \in V_i && (8) \\ (\frac{\partial L_r}{\partial y_i}(s.p), \phi_i) &= 0 \quad \forall \phi_i \in V_i && (9) \\ (\frac{\partial L_r}{\partial v_i}(s.p), v_i - u_i) &\geq 0 \quad \forall v_i \in U_{ad}^i && (10) \\ (\frac{\partial L_r}{\partial \lambda_{ij}}(s.p), d\lambda) &= 0 \quad \forall \ d\lambda \in H^{-\frac{1}{2}}(\sigma_{ij}), \ i \neq j && (11) \\ (\frac{\partial L_r}{\partial \omega_{ij}}(s.p), d\omega) &= 0 \quad \forall \ d\omega \in H^{-\frac{1}{2}}(\sigma_{ij}), \ i < j && (12) \\ (\frac{\partial L_r}{\partial q_{ij}}(s.p), dq) &= 0 \quad \forall \ dq \in H^{\frac{1}{2}}(\sigma_{ij}), \ i < j && (13) \end{aligned}$$

[2] We use short expression of the L_r variables: $(v_i, y_i, p_i, \omega_{ij}, \lambda_{ij}, q_{ij})$ instead of $((v_i)_{1\leq i\leq m}, (y_i)_{1\leq i\leq m}, (p_i)_{1\leq i\leq m}, (\omega_{ij})_{1\leq i<j\leq m}, (\lambda_{ij})_{1\leq i\neq j\leq m}, (q_{ij})_{1\leq i\neq j\leq m})$.

Let (u, y, p) be the saddle point of the Lagrangian associated with the problem (1). From (8) and (11), it follows that: $y_i = y_j$ and $\frac{\partial y_i}{\partial \eta_{ij}} = \frac{\partial y_i}{\partial \eta_{ij}}$ on σ_{ij}. So, y_i is exactly the restriction of y on Ω_i.

As for p_i, from (9), (12) and (13), we obtain both the continuity and the flux continuity of the adjoint state. Thus, $p_i = p/\Omega_i$.

Finally, using the optimality of u_i (10) and the equation for p_i, we deduce $(p_i + \nu u_i, v_i - u_i)_{L^2(\Omega_i)} \geq 0$, $\forall v_i \in U_{ad}^i$ and get the optimality condition in the global domain.

Solution Algorithm

Due to the above proposition, we have just to search a saddle point of L_r. So, we propose a modification of the algorithm ALG3 in [GL89b]:

Algorithm

Step 1. Initialization: u_i^1 given in $L^2(\Omega_i)$, $(\omega_{ij}^1)_{i<j}$ and $(\lambda_{ij}^1)_{i\neq j}$ given in $H^{-\frac{1}{2}}(\sigma_{ij})$ and $(q_{ij}^0)_{i<j}$ given in $H^{\frac{1}{2}}(\sigma_{ij})$.

Step 2. Iteration: For $n = 1, 2, ...$, compute y_i^n and p_i^n such that $y_i^n \in V_i$, $p_i^n \in V_i$ and

$$\begin{cases} -\Delta y_i^n &= f_i + u_i^n \quad \mathit{in} \quad \Omega_i, \\ \frac{\partial y_i^n}{\partial \eta} &= \omega_{ij}^n \qquad \mathit{on} \quad \sigma_{ij},\ j \neq i \end{cases}$$

$$\begin{cases} -\Delta p_i^n &= y_i^n - y_d^i \qquad\qquad \mathit{in} \quad \Omega_i \\ \frac{\partial p_i^n}{\eta_{ij}} &= \lambda_{ij}^n + r(y_i^n - q_{ij}^{n-1}) \quad \mathit{on} \quad \sigma_{ij},\ j \neq i \end{cases}$$

- Compute the gradient g_i^n of $L_r(v_i, ...)$, $g_i^n = p_i^n + \nu u_i^n$
- $u_i^{n+1} = u_i^n + t^n d_i^n$, $t^n > 0$ minimization along the descent direction d_i^n computed from g_i^n by BFGS formula.
- Update the Lagrange multipliers and q_{ij}^n: $\rho^n > 0$, $\rho_\omega^n > 0$

$$\begin{aligned} \omega_{ij}^{n+1} &= \omega_{ij}^n + \rho_\omega^n (p_i^n - p_j^n) \\ \lambda_{ij}^{n+\frac{1}{2}} &= \lambda_{ij}^n + \rho^n (y_i^n - q_{ij}^{n-1}) \\ 2rq_{ij}^n &= (\lambda_{ij}^{n+\frac{1}{2}} + \lambda_{ji}^{n+\frac{1}{2}}) + r(y_i^n + y_j^n) \\ \lambda_{ij}^{n+1} &= \lambda_{ij}^{n+\frac{1}{2}} + \rho^n (y_i^n - q_{ij}^n) \end{aligned}$$

Remark 3 *In the above algorithm, eliminating λ_{ij}^n and q_{ij}^n for the case where $\rho^n = \rho_\omega^n = r$, the p_i-boundary conditions on σ_{ij} are transformed into*

$$-\frac{\partial p_i^{n+1}}{\partial \eta_{ij}} + r y_i^{n+1} = \frac{\partial p_j^n}{\partial \eta_{ji}} + r y_j^n \ \mathit{on}\ \sigma_{ij}.$$

These are precisely the conditions of algorithm Alg2 in [Ben94], where transmission conditions of [Lio90] are applied to the optimal control problem. The y_i-boundary conditions on σ_{ij} are written as

$$-\frac{\partial y_i^{n+1}}{\partial \eta_{ij}} + r p_i^n = \frac{\partial y_j^n}{\partial \eta_{ji}} + r p_j^n \ \mathit{on}\ \sigma_{ij}$$

which are close (but not identical) to those given in Alg2.

3 Numerical Results

As a test example, we consider a boundary optimal control problem given in [BGL73] defined by its direct state equation:

$$\left\{\begin{array}{rcll} -\Delta y(v) & = & f & in \quad \Omega, \\ y(v) & = & 0 & on \quad \Gamma_3 \cup \Gamma_4, \\ \frac{\partial y(v)}{\partial \eta} & = & v & on \quad \Gamma_1 \cup \Gamma_2. \end{array}\right. \tag{14}$$

and we want to minimize over $U_{ad} = L^2(\Gamma_1 \cup \Gamma_2)$, the function

$$J(v) = \frac{1}{2}(\int_\Omega (y(v) - y_d)^2 dx + \nu \int_{\Gamma_1 \cup \Gamma_2} v^2 d\sigma).$$

In the computations, we take $\Omega =]0,4[\times]0,1[$, $f(x,y) = 2(-x^2 - y^2 + 4x + y)$ and $y_d(x,y) = (y - y^2)(x^2 - 4x) - 8\nu$, for $(x,y) \in \Omega$.

We split the domain Ω into m subdomains (see Fig. 1). A finite difference scheme is used to discretize both the direct and the adjoint state systems. We run the proposed algorithm with $\rho^n = \rho^n_\omega = r$. The descent direction is computed from the gradient vector of the Lagrangian L_r with respect to the control, the line search process and BFGS formula are carried out similarly to $M1QN3$ algorithm [GL89a]. The iterations of the algorithm are stopped when the norm of the gradient is small enough. For $\nu = 1.25$, $r = 0.85$ and for different values of h (discretization step) and m (subdomain number), the relative L^2-error on the direct state and the interface error average are worked out. In Table 1, we define

$$err_y = \left(\frac{\sum_{i=1}^m \|y_i^n - ye_i\|^2_{L^2(\Omega_i)}}{\sum_{i=1}^m \|ye_i\|^2_{L^2(\Omega_i)}}\right)^{\frac{1}{2}}, \; err_{yij} = \frac{1}{m}\sum_{i=1}^m \sum_{j>i,\ \sigma_{ij}\neq\emptyset} \|y_i^n - y_j^n\|_{L^2(\sigma_{ij})}$$

where y_i^n is the computed solution on Ω_i at the nth iteration when the stopping criterion is attained and $ye_i = ye/\Omega_i$ with $ye(x,y) = (y^2 - y)(x^2 - 4x),\ for(x,y) \in \Omega$, is the analytic solution to the global problem. For $m = 1$, the minimization of J is done by the $M1QN3$ algorithm. Table 1 shows that the domain splitting into two subdomains does not affect much the convergence results of our algorithm and the y_i connect very well on the interface because of the symmetry of this problem. The convergence is attained in few iterations and depends on the '*quasi-optimal*' choice of r.

4 Conclusion

Using domain decomposition, the original optimal control problem is transformed into a saddle point problem. We have used augmented Lagrangian techniques studied by Fortin and Glowinski[FG82], Glowinski and Le Tallec[GL89b]. We have combined a descent method with a multipliers one. The proposed algorithm gives different ways of dealing with constraints on interfaces. Moreover, it is well suited to parallel processors.

Figure 1 The decomposed domain of the test example ($m = 8$).

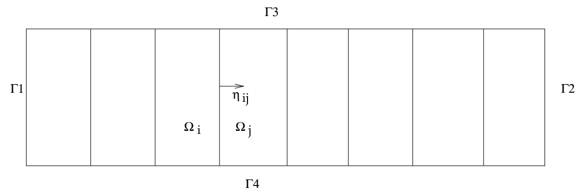


Table 1 Direct State Results. h: step discretization, m: subdomain number.

h^{-1}	m	err_y	err_{yij}
16	1	1.2226217904E-06	- - -
	2	3.5817536178E-05	1.5060315952E-06
	4	1.4511406491E-02	1.5495620667E-02
	8	8.7904304272E-02	8.6599163711E-02
64	1	1.0268570911E-06	- - -
	2	1.3455085083E-05	1.3616281608E-06
	4	1.1562970614E-02	1.5400098264E-02
	8	7.9211773115E-02	8.0130822639E-02

Acknowledgement

This work was carried out within the IDOPT project joint to CNRS, INRIA, Joseph Fourier University (Grenoble I) and Polytechnic National Institute of Grenoble and the computations were done on the parallel machine SP1 of IMAG. The author is grateful to Prof. J. Blum and B. Luong for many valuable discussions and suggestions related to this work and wishes to thank J. C. Gilbert and C. Lemaréchal for the use of their efficient minimization algorithm ($M1QN3$ routine, INRIA MODULOPT library).

REFERENCES

[Ben94] Benamou J.-D. (April 1994) Domain decomposition methods with coupled transmission conditions for the control of systems governed by elliptic partial differential equations. Technical Report 2246, INRIA.

[BGL73] Bensoussan A., Glowinski R., and Lions J.-L. (May 1973) Méthode de décomposition appliquée au contrôle optimal de systèmes distribués. *Lecture Notes in Computer Science* 5. (In the fifth IFIP Conference on optimization techniques).

[FG82] Fortin M. and Glowinski R. (1982) *Méthodes de Lagrangien augmenté, Application aux problèmes aux limites.* Bordas.

[GL89a] Gilbert J.-C. and Lemaréchal C. (1989) Some numerical experiments with variable storage quasi-newton algorithms. *Mathematical Programming* 45: 407–435.

[GL89b] Glowinski R. and Le Tallec P. (1989) *Augmented Lagrangian and Operator Splitting in Nonlinear Mechanics.* SIAM.

[GL90] Glowinski R. and Le Tallec P. (1990) Augmented Lagrangian interpretation of the nonoverlapping Schwarz alternating method. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Proc. 3rd Conference on Domain Decomposition Methods.*, pages 224–231. SIAM, Philadelphia.

[Lio68] Lions J.-L. (1968) *Contrôle optimal de systèmes gouvernés par des équations aux dérivées partielles.* Dunod, Paris.

[Lio90] Lions P.-L. (1990) On the Schwarz alternating method III:
A variant for nonoverlapping subdomains. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Proc. 3rd Conference on Domain Decomposition Methods.*, pages 202–223. SIAM, Philadelphia.

34

Decoupling and Modal Synthesis of Vibrating Continuous Systems

Frédéric Bourquin and Rabah Namar

Laboratoire des Matériaux et des Structures du Génie Civil,
UMR113 LCPC/CNRS, 2 allée Kepler, 77420 Champs sur
Marne, France

1 Introduction

The modal synthesis of a structure that is decomposed into substructures is a Rayleigh-Ritz approximation of the global eigenvalue problem over a space spanned by a few eigenmodes of each substructure in addition to some functions, called *coupling modes* describing the interfacial displacements. Different methods based on intrinsic choices of such *coupling modes* are presented. In particular, extension operators from the boundary of each subdomain to the whole interface are introduced in view of defining, both in a continuous and in a discrete setting, "generalized" Neumann-Neumann preconditioners, the eigenfunctions of which are used as *coupling modes* for approximating the global eigenvalue problem.

Let us consider the model problem of a vibrating three-dimensional body Ω. The family of eigenpairs $\{\lambda_k, \boldsymbol{u}_k\}_{k=1}^{+\infty}$, arranged in nondecreasing order of the eigenvalues λ_k, solve the eigenvalue problem

$$\begin{array}{rcll} -div(\mathcal{A}\boldsymbol{e}(\boldsymbol{u})) &=& \lambda\rho\boldsymbol{u} & \text{in } \Omega \\ \boldsymbol{u} &=& 0 & \text{on } \partial\Omega, \end{array} \tag{1.1}$$

where $\boldsymbol{e}(\boldsymbol{u})$ denotes the linearized strain tensor associated with the displacement field $\boldsymbol{u}$, $\mathcal{A}$ the tensor of elastic moduli, ρ the mass density, and $\partial\Omega$ the boundary of Ω. The structure is assumed to be clamped for the sake of simplicity, but any set of boundary conditions yielding a symmetric variational formulation like traction free, mixed, or third kind boundary conditions may lead to the same conclusions. Other models like membrane or plate models can also be considered.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

The domain Ω is partitioned into p nonoverlapping subdomains Ω_1,..., Ω_p, which are separated by an interface Γ. For modal synthesis with overlap see [CDVM96, CDVM95]. The fixed interface eigenpairs $\left\{\lambda_j^i, \boldsymbol{u}_j^i\right\}_{j=1}^{+\infty}$ of subdomain Ω_i solve the problem

$$\begin{array}{rcll} -div(\mathcal{A}\mathbf{e}(\mathbf{u})) & = & \lambda\rho\mathbf{u} & \text{in} \quad \Omega_i \\ \mathbf{u} & = & 0 & \text{on} \quad \partial\Omega_i, \end{array} \tag{1.2}$$

The eigenfunctions $\mathbf{u}_j^i$ are extended by zero outside Ω_i and are thus defined on the whole domain Ω. Therefore, these so-called *fixed interface modes* can be used as trial functions in a Rayleigh-Ritz approximation of problem (1.1.1). Of course, they must be supplemented by a family of trial functions $\mathbf{w}_{\Gamma\ell}$ that do not identically vanish on the interface Γ between the subdomains. We call them *coupling modes.* The basic modal synthesis method thus amounts to a Galerkin approximation of problem (1.1.1) over the space

$$V_N = Span \left\{ \bigcup_{i=1}^{p} \left(\mathbf{u}_j^i\right)_{j=1}^{N_i} \bigcup \left(\mathbf{w}_{\Gamma\ell}\right)_{\ell=1}^{N_\Gamma} \right\} \tag{1.3}$$

for some numbers N_i, $1 \leq i \leq p$, and N_Γ. Therefore, in its original version, modal synthesis is not an iterative algorithm, in contrast to those proposed in [Mal92, Mal96, SC96, CL96, Lui96, dV96], but rather a method of approximation. However, the approximate eigenpairs may be enhanced by postprocessing as in [Cha83] or more specifically in [Bal96], and generally speaking they may serve as good starting points for iterative domain decomposition correction algorithms.

Modal synthesis methods have been introduced in aerospace engineering in the sixties in order to save memory storage when analyzing the dynamics of large structures. These methods are now used in particular by nuclear, off-shore, automobile, and aerospace industries. The advantages of such methods are potentially numerous. Of course, they are amenable to parallel implementation. This point will be made clearer in the third and fourth sections. Furthermore, they can include experimental measurements on the substructures. Moreover, parametric studies involving local perturbations in view of sensitivity analysis or reanalysis can be performed cheaply [Tra96]. Finally, the substructuring concept extends to fluid-structure interaction [MO79], soil-structure interaction [Clo93], and buckling [Val82]. There exists a wide variety of methods, depending on the boundary conditions imposed to each substructure, and on the coupling strategy. In particular, many hybrid methods have been proposed and discussed; see, e.g., [MN71, Des89, DO96, J́85, Tra92b, Tra92a]. They correspond to other kinds of boundary conditions for the definition of the local modes. To a large extent, their numerical analysis is open. They fall in the general class of non-conforming methods since, at the continuous level at least, the continuity across the interface cannot be imposed if for example the local modes are associated with boundary conditions of Neumann type along the interface. General expositions on modal synthesis can be found in [Imb79, Mei80, Cra85, J́85, Mas88, Gib88, Tra92b].

This paper aims first at reviewing *a priori* error estimates for such methods and second at exploring new coupling strategies. In the next section, we revisit the pioneering work of [Hur65]. The third section is devoted to more recent methods involving *coupling modes* defined as the eigenfunctions of the Poincaré-Steklov operator [Bou92, Bd92b, Bd92a]. Since the computation of these *coupling modes* may be expensive when the interface is large and complex, cheaper *coupling modes* are introduced in section four. To this end, new operators of Neumann-Neumann type are defined and put in vibration. They are based on special extension operators from the boundary of each subdomain to the whole interface. Numerical tests confirm the accuracy of the resulting modal synthesis method. We close with a few comments.

2 Hurty's method

Let V denote the space of global test functions and V_Γ the space of their restrictions to the interface Γ, namely $V = H_0^1(\Omega)$ and $V_\Gamma = H_{00}^{1/2}(\Gamma)$ if Dirichlet boundary conditions are imposed on $\partial\Omega$ as in (1.1.1). In a continuous setting, Hurty's method (see [Hur65] and [CB68]) would amount to choose the *coupling modes* as the "harmonic" extensions to each subdomain of all elements of a given basis of V_Γ.

Now, if N_i *fixed interface modes* are retained to describe the dynamics of subdomain Ω_i, and if $\lambda_k^{HCB,N}$ denotes the k^{th} eigenvalue resulting from Hurty's procedure, the following error bound is derived in [Bou92]:

$$0 \leq \lambda_k^{HCB,N} - \lambda_k \leq \sum_{i=1}^{p} C_i(1+N_i)^{-1}. \tag{2.1}$$

The constants C_i depend on k but not on N_i. Similar error bounds hold for the eigenfunctions in L^2-norm as well as in energy norm. For two-dimensional elasticity, and for bars under traction [Bou90], the estimate would behave like $N_i^{-3/2}$, and N_i^{-3} respectively. Note that the rate of convergence deteriorates when the dimension of the problem increases.

The constants C_i in (1.2.1) behave as follows: $C_i \sim \left(\frac{\rho_i}{E_i}\right)^{3/2} Vol(\Omega_i)$, if ρ_i and E_i stand for the mass density and Young's modulus of substructure Ω_i, when these quantities are constant on Ω_i. For two-dimensional elasticity, we obtain $C_i \sim \left(\frac{\rho_i}{E_i}\right)^{3/2} (Vol\,(\Omega_i))^{3/2}$. Combining above estimates yields a rational way to choose the number of *fixed interface modes* of each substructure relative to the others.

The optimality of the error bounds is highlighted by the numerical experiments presented in [Bd92b]: the true error behaves like $N^{-3/2}$ for the two-dimensional membrane problem which has the same properties as plane elasticity from the viewpoint of modal synthesis. This optimality is also suggested by the proof of the error bound which is obtained as the rest of a series expansion that converges no faster than indicated. The key ingredient of the error analysis is real interpolation theory in Sobolev spaces [LM68]. In order to explain in a simple way where the exponents come from, let us consider the Fourier series expansion on the basis $(\sin j\pi x)_{j=1}^{+\infty}$ of the function 1 over [0,1]: $1 = \sum_{j=1}^{+\infty} \alpha_j \sin j\pi x$. Of course $\sum_{j=1}^{+\infty} (\alpha_j)^2 < \infty$. However, if $\nu_j = j^2\pi^2$, then $\sum_{j=1}^{+\infty} \nu_j \,(\alpha_j)^2 = +\infty$ otherwise the function 1 would vanish at both ends of the interval, but $\sum_{j=1}^{+\infty} (\nu_j)^{1/2-\varepsilon} (\alpha_j)^2 < \infty \; \forall \; \varepsilon > 0$. This property extends *mutatis mutandis* to arbitrary n-dimensional domains as a consequence of [LM68] interpolation theory. Weyl's formula proves also useful to derive the error bound. Working directly at the level of the finite element discretization is possible, but does not yield optimal error bounds because the underlying PDE is hidden. Notice that different error bounds can be derived for plates [Bd92a].

In order to compute the final generalized mass and stiffness matrices, all the modes are usually discretized with finite elements. If h denotes the discretization parameter, and $\lambda_k^{HCB,N,h}$ the k^{th} eigenvalue computed with Hurty's method, then the error $\lambda_k^{HCB,N,h} - \lambda_k$ can be estimated by the sum of the right-hand side of (1.2.1), and a discretization error Ch^β, for some constant $C(N)$, and some $\beta > 0$ which depends on the finite element method and on the smoothness of the *fixed interface modes* [Bou92]. A variant of this method enables one to use incompatible meshes on the different subdomains [FG94]. See [RTG96] for a comparison in the frequency domain of the continuous version of the modal synthesis method with the associated discrete version.

In general Hurty's method combines F.E. discretization and local mode truncation in a way that prevents one from using it when the interface contains many degrees of freedom, because the resulting mass and stiffness matrices are still quite large and dense, therefore difficult to handle. Even forming these matrices proves time-consuming since the Schur complement matrix has to be computed. A possible solution is to avoid computing these matrices and to use an iterative solver within the eigenvalue solver for the final generalized eigenvalue problem. An other strategy consists of further reducing the size of this final eigenvalue problem. In this direction, *coupling modes* can be defined at both the continuous level and the discrete level such that a prescribed accuracy of modal synthesis is achieved by means of a given number of them that does not depend on the mesh size [Bou89]. We define them in the next section.

3 A Poincaré-Steklov Operator-based Method

Let $\mathbf{u}$ and $\mathbf{v}$ denote any displacement fields in V_Γ, resulting in "harmonic" (in the sense of elasticity) extensions $\tilde{\mathbf{u}}^i$ and $\tilde{\mathbf{v}}^i$ on every subdomain Ω_i. The bilinear form $b(\mathbf{u},\mathbf{v}) = \sum_{i=1}^{p} \int_{\Omega_i} \sigma(\tilde{\mathbf{u}}^i) : e(\tilde{\mathbf{v}}^i)$ defines a scalar product over V_Γ and the associated isomorphism T satisfies $T\mathbf{u} = \sum_{i=1}^{p} \left(\left(\mathcal{A}_i e \left(\tilde{\mathbf{u}}^i \right) \right) .\boldsymbol{n}^i \right)_{|\Gamma \cap \partial\Omega_i}$, if $\boldsymbol{n}^i$ denotes the unit outer normal vector to Ω_i along $\partial\Omega_i$, and $\mathcal{A}_i = \mathcal{A}_{|\Omega_i}$. This well-known Poincaré-Steklov operator is of course the continuous counterpart of the Schur complement matrix.

From standard spectral theory, the problem *find* $(\lambda, \mathbf{u}) \in \mathrm{I\!R} \times V_\Gamma$ *such that*

$$b(\mathbf{u},\mathbf{v}) = \lambda \int_\Gamma \mathbf{u}\mathbf{v} \quad \forall \mathbf{v} \in V_\Gamma \tag{3.1}$$

is well-posed and admits a family of solutions $\{\lambda_{\Gamma\ell}, \mathbf{u}_{\Gamma\ell}\}_{\ell=1}^{+\infty} \in \mathrm{I\!R} \times V_\Gamma$ such that $(\mathbf{u}_{\Gamma\ell})_{\ell=1}^{+\infty}$ forms a basis of V_Γ. The "harmonic" extension of $\mathbf{u}_{\Gamma\ell}$ to each component Ω_i is continuous across the interface Γ and is thus defined on the whole domain

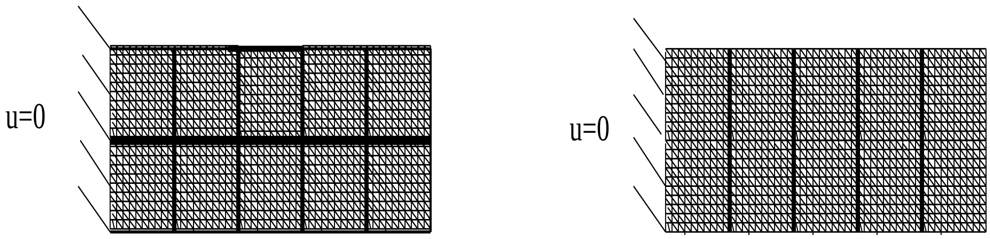


Figure 1 A two-dimensional elastic beam, clamped on the left side and free everywhere else, decomposed in two ways.

as a trial function $\tilde{\mathbf{u}}_{\Gamma\ell} \in V$ which we take as *coupling mode*. The Rayleigh-Ritz approximation of the global eigenvalue problem based on N_i *fixed interface modes* from each substructure as in Hurty's method and N_Γ such *coupling modes* yield approximate eigenvalues $\lambda_k^{PS1,N}$. For three-dimensional elasticity, the error bound

$$0 \leq \lambda_k^{PS1,N} - \lambda_k \leq \sum_{i=1}^{p} C_i {N_i}^{-1} + C_\Gamma {N_\Gamma}^{-\alpha} \tag{3.2}$$

has been derived in [Bou89, Bou91]. In (1.3.2), α denotes the exponent of the most severe vertex or edge singularity of the local source problems with homogeneous Dirichlet conditions on the interface. For two-dimensional elasticity, we get the exponent 2α instead of α. See also [Bd92a] for plates. That the rate of convergence does not depend directly on the smoothness of the eigenmode is an interesting feature. For the approximation of *coupling modes*, we refer to [Bou92], and also to [BVPA94]. The discrete *coupling modes* can be computed by a Lanczos method combined with the Neumann-Neumann algorithm of [BGLT88], as in [Bd92b] without forming the Schur complement matrix, or as in [CL96] where the descent directions of the inner loop are stored in view of solving more efficiently the source problem $T\mathbf{u} = \mathbf{g}$ with successive right-hand sides.

This basic Poincaré-Steklov operator based modal synthesis method proves very accurate because the first few global eigenpairs are correctly described by means of a small number of *fixed interface* and *coupling modes* [Bd92b]. Moreover, it can be parallelized in the same way as the Neumann-Neumann algorithm applied to source problems.

Although taking advantage of the identity matrix along Γ instead of the consistent mass matrix associated with the scalar product $\int_\Gamma \mathbf{uv}$ apparently does not deteriorate too much the accuracy of the whole procedure, one may get rid of the possibly non standard implementation of this mass matrix by using a non-local inertia along Γ [BN97a].

4 Extended Neumann-Neumann Preconditioners

Now, can one speed up the method by defining new *coupling modes* that would be much cheaper to compute, and that would yield comparable accuracy? A natural idea is to replace the Poincaré-Steklov operator T by a preconditioner, like the Neumann-Neumann preconditioner introduced in [BGLT88]. This operator S is defined as

$$S : V_\Gamma' \longrightarrow V_\Gamma, \quad \mathbf{v} \longrightarrow S\mathbf{v} = \sum_{i=1}^{p} P_i S_i R_i \mathbf{v}, \tag{4.1}$$

where $P_i : W_i = tr_{|\Gamma\cap\partial\Omega_i}(V) \longrightarrow V_\Gamma$ is a continuous extension operator, $R_i : V_\Gamma' \longrightarrow W_i'$ is a continuous restriction operator and $S_i : W_i' \longrightarrow W_i$ is the Neumann-to-Dirichlet operator associated with the subdomain Ω_i and its boundary, that is to say $S_i\mathbf{u} = tr_{|\Gamma\cap\partial\Omega_i}(\tilde{\mathbf{u}}^i)$, where

$$\begin{array}{rcl} -div\mathcal{A}_i e(\tilde{\mathbf{u}}^i) + d\tilde{\mathbf{u}}^i &=& 0 \text{ in } \Omega_i, \\ \mathcal{A}_i e(\tilde{\mathbf{u}}^i).\boldsymbol{n}_i &=& \mathbf{u} \text{ on } \Gamma\cap\partial\Omega_i, \\ \tilde{\mathbf{u}}_i &=& 0 \text{ on } \partial\Omega_i\cap\partial\Omega, \end{array} \tag{4.2}$$

for some $d > 0$. In order to ensure the symmetry of S, we shall always choose $R_i = P_i^t$. We have the following

Proposition: *let us set* $E_i = P_i P_i^t$, *and* $E = \sum_{i=1}^{p} E_i$. *Then*

i) $\exists C{>}0$, *such that* ${}_{V_\Gamma}\langle Sv, v\rangle_{V_\Gamma'} \leq C\|v\|^2_{V_\Gamma'}$, $\forall v \in V_\Gamma'$.

ii) $Ker(S) = Ker(E)$ *and* $Ker(E) = \bigcap_{i=1}^{p} Ker(E_i)$.

iii) The direct sum of the eigenspaces of S associated with strictly positive eigenvalues and of $Ker(E)$ *is dense in* V_Γ' *and in* $L^2(\Gamma)$.

Proof: i) is a direct consequence of the continuity of all operators involved in definition (1.4.1). On the other hand, the coerciveness of each operator S_i which is due to the positivity of d, leads to the inequality ${}_{V_\Gamma}\langle Sv, v\rangle_{V_\Gamma'} \geq \alpha \sum_{i=1}^{p} \|P_i^t v\|^2_{W_i'}$, for some positive α. Since ${}_{V_\Gamma}\langle Ev, v\rangle_{V_\Gamma'} = \sum_{i=1}^{p} \|P_i^t v\|^2_{W_i'}$, we also have the opposite inequality. Moreover, each operator E_i is non-negative, hence ii). Finally iii) follows from the symmetry and compactness of S on $L^2(\Gamma)$ and from ii).

Interface without Cross-points

Consider a domain decomposed in p slices separated by edges Γ_j that of course do not intersect. In this case, the space V_Γ coincides with $\prod_{j=1}^{p-1} tr_{|\Gamma_j}(V)$ and the extension operators $P_i : W_i \longrightarrow V_\Gamma$, defined by

$$P_i\mathbf{u} = \left\{ \begin{array}{lll} \mathbf{u} & \text{on} & \partial\Omega_i \cap \Gamma, \\ 0 & \text{on} & \Gamma - (\partial\Omega_i \cap \Gamma), \end{array} \right. \tag{4.3}$$

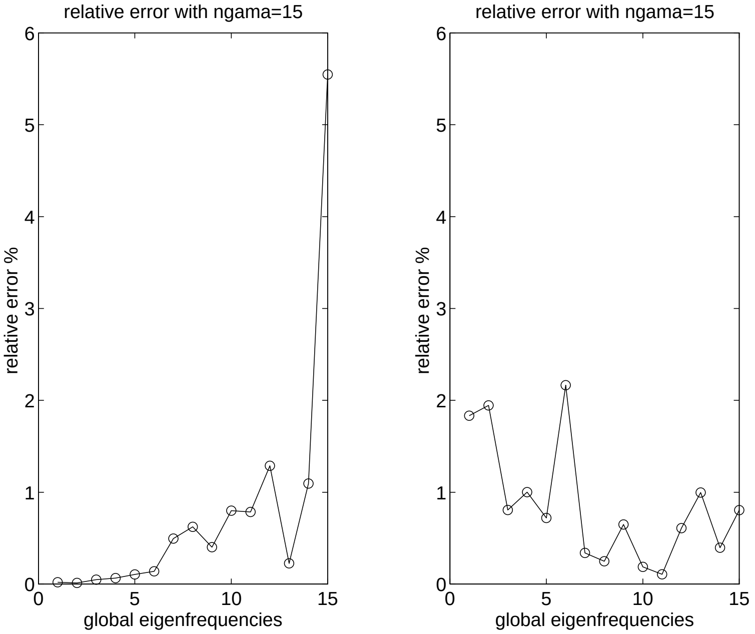


Figure 2 Interface *without* cross-points (5 slices): Relative accuracy on the 15 smallest global eigenfrequencies for the standard Poincaré-Steklov based method (left) with $N_\Gamma = 15$ and for the Neumann-Neumann based method (right) with $N_\Gamma = 15$

are continuous. The adjoint operators R_i coincide with the usual restrictions to $\partial\Omega_i \cap \Gamma$. The operator S is well defined at the continuous level. It has no kernel and possesses a countable family of eigenpairs $\left(\lambda^{\Gamma\ell}, \mathbf{u}^{\Gamma\ell}\right)_{\ell=1}^{\infty}$ arranged such that the eigenvalues decrease towards zero. The "harmonic" extensions to each component of the eigenfunctions $\mathbf{u}^{\Gamma\ell}$ can be chosen as *coupling modes*. This idea is close to the concept underlying [BKP95]. The resulting modal synthesis method proves less accurate than the original one, but the new *coupling modes* $\mathbf{u}^{\Gamma\ell}$ can be computed quite rapidly through a Lanczos method (see e.g. [CL96] and included references), because now each step mainly consists of computing $S\mathbf{g}$ for a given $\mathbf{g}$. There are no more internal loops for solving any source problem on Γ. The accuracy of the resulting modal synthesis method is very encouraging as shown on fig 2 since it yields a similar accuracy as the Poincaré-Steklov based method at a much smaller cost.

When the interface exhibits a more complex geometry than before, extension by zero is no more possible if $H^{1/2}(\Gamma)$ regularity is to be preserved for the *coupling modes*. However, an extension of the method can be designed.

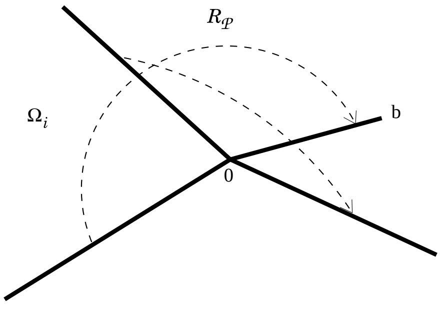


Figure 3 How the extension operator works

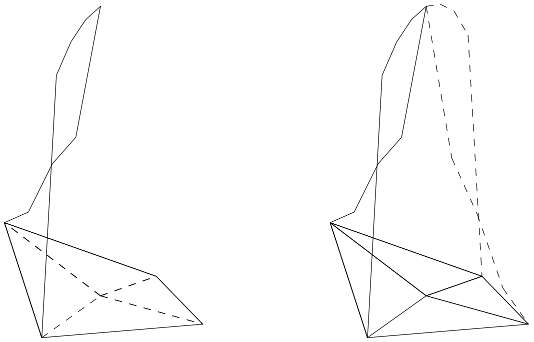


Figure 4 A function defined on the boundary of a subdomain and its extension to the whole interface

Interface with cross-points

Some preliminaries are needed: let f denote a continuous function on the interval $[a, 0],\ a < 0,$ to be extended to some interval $[0, b],\ b > 0,$ then define the function

$$({}^R\mathcal{P}f)(x) = \begin{cases} f(x) & \text{on } [a, 0], \\ f(\frac{a}{b}x)\text{'}(x) & \text{on } [0, b], \end{cases} \tag{4.4}$$

where ' denotes a smooth cut-off function vanishing in the vicinity of b. The operator ${}^R\mathcal{P}$ enjoys $H^{1/2}$ continuity from interpolation theory. This generic reflection operator can be put to work in order to define the extension operators ${}^RP_i$, as follows: if $\mathbf{u}$ stands for a function defined over $\Gamma \cap \partial\Omega_i,$ we want to extend it in a continuous way to the adjacent edges. Define a curvilinear abscissa over the edges of Γ sharing a given vertex which is a cross-point. For such an edge not belonging to $\partial\Omega_i,$ parametrized by $s \in [0, b],$ choose such an edge belonging to $\partial\Omega_i,$ parametrized by $s \in [a, 0],$ and apply the operator ${}^R\mathcal{P}$ defined above. Repeat this for all edges adjacent to $\Gamma \cap \partial\Omega_i$ (see Figures 3 and 4). Of course, the choice of the edge parametrized by $s \in [a, 0]$

is somewhat arbitrary and a weighted average of operators ${}^R\mathcal{P}$ corresponding to different edges of $\Gamma \cap \partial\Omega_i$ can also be used. In any case, we end up with an operator ${}^RP_i$ for every subdomain, and operators ${}^RS$ and ${}^RE$ as in (1.4.1) and in the proposition. They are defined in a continuous setting as well as in a discrete setting. We omit the details of the discretization and of the implementation here. Although the perfect locality of the original Neumann-Neumann preconditioner is lost, some locality is preserved because of the cut-off function '. The kernel of such operators are not reduced to $\{0\}$ in general, but the proposition suggests to choose as *coupling modes* the first few eigenfunctions of ${}^RS$ associated with its largest eigenvalues in addition to $Ker({}^RE)$. At the discrete level, this kernel is easy to compute since the matrices ${}^RE_i ={}^R P_i{}^RP_i^t$ only depend on the mesh and on the geometry of the interface. No subdomain solve is required. From ii) of the proposition, the kernel can be computed as the intersection of the local kernels $Ker({}^RE_i)$, that will be computed in parallel with minimal data exchange between subdomains. Then the Rayleigh-Ritz approximation of the eigenvalue problem ${}^REx = \mu x$ over $\bigcup_{i=1}^p Ker({}^RE_i)$ is performed. The rank of the corresponding Rayleigh matrix may be maximal. In this case, $Ker({}^RE) = \{0\}$. However, the smallest eigenvalue of this matrix can be very small. Therefore, a variant of this modal synthesis method consists of computing several of the smallest eigenvalues of ${}^RE$ and corresponding eigenvectors that will supplement the set of eigenvectors of ${}^RS$. In particular, it may be interesting to keep as *coupling modes* an independent set of vectors in $\bigcup_{i=1}^p Ker({}^RE_i)$, in view of parallel implementation. On the other hand, $Ker({}^RE)$ may be very large, therefore a strategy to filter out unwanted, highly oscillating functions of this kernel should be put to work in this case.

This method, referred to as the R-method, has been tested on the reference structure and the relative error on the eigenfrequencies is reported on Figure 5. It compares favourably with the standard Poincaré-Steklov based method since a similar accuracy is obtained at a smaller cost.

A conceptually simpler variant of above extension operators is based on the linear extension operator defined as

$$({}^L\mathcal{P}f)(x) = \left\{ \begin{array}{ll} f(x) & \text{on } [a,0], \\ f(0)(1-\frac{x}{b}) & \text{on } [0,b], \end{array} \right. \tag{4.5}$$

that is well-defined at the discrete level but not at the continuous level because functions in $H^{1/2}$ do not have traces. In this case, we noticed that the associated operators ${}^LE_i$ have no kernel. Results of poorer quality than with the R-method are obtained for this variant, referred to as L-method, as shown on Figure 6.

5 Concluding Remarks

i) According to recent numerical experience of the authors, the Poincaré-Steklov based methods are the most accurate at least at very low frequency. The R-method proves slightly less accurate but a better cost effectiveness is expected. It turns out that the L-method which is not defined at the continuous level is less accurate than the other

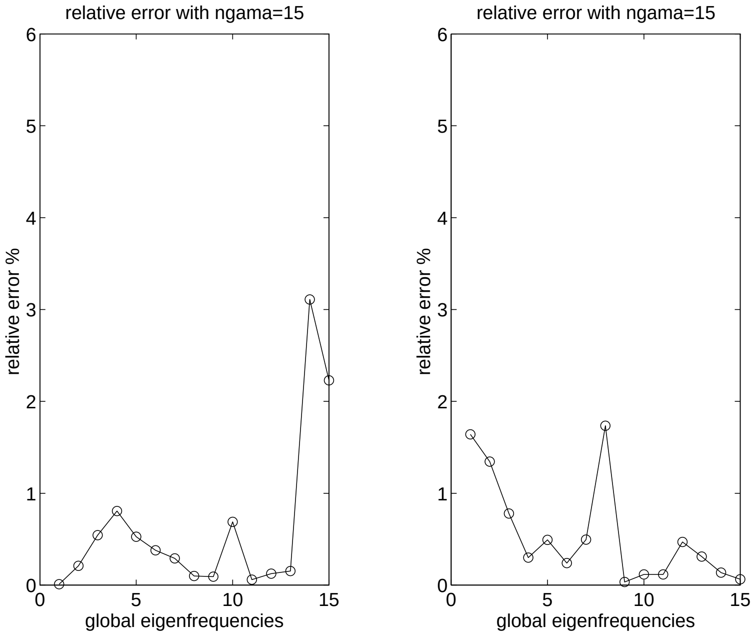


Figure 5 Interface with cross-points (10 subdomains): relative accuracy on the 15 smallest global eigenfrequencies for the standard Poincaré-Steklov based method (left) for $N_\Gamma = 15$ and for the R-method (right) with $N_\Gamma = 15$, and with 1/5 of the eigenvectors of ${}^RE$ over $\bigcup_{i=1}^{p} Ker({}^RE_i)$

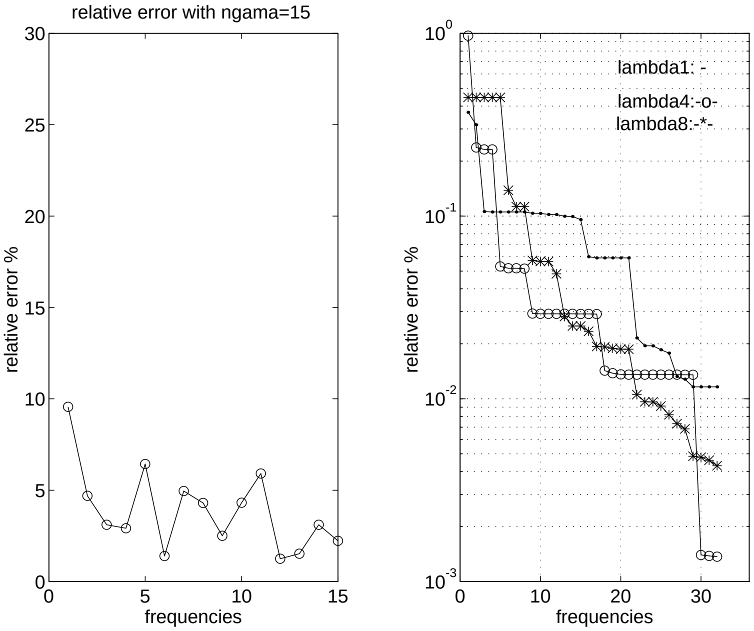


Figure 6 Interface with cross-points (10 subdomains), L-method. Relative accuracy on the 15 smallest global eigenfrequencies (left) for $N_\Gamma = 15$, and decay (right) w.r.t. N_Γ of the relative error on the global eigenfrequencies 1, 4 and 8.

ones. All of them are amenable to parallel computing. The R- and L-methods involve slightly more communication efforts between subdomains.

ii) Variants of the R- or L-method can be designed. In particular, parametrized families of extension operators can be introduced in view of optimizing the accuracy of the method. A detailed version of section 1.4 on the design and use of extended Neumann-Neumann preconditioners will be available in [BN97a].

iii) Extension to three-dimensional problems seems conceptually possible. However, from the practical point of view, defining the extension operator from the boundary of each subdomain to the whole set of interfacial faces may appeal to geometrical data that are not directly available from the numerical description of each subdomain.

iv) On the other hand, the case of plate-like problems can be treated by means of similar but more complicated techniques. Here, extension operators that preserve $H^{3/2} \times H^{1/2}$ continuity along the interface must be designed [BN97b].

v) It would be interesting to combine the Poincaré-Steklov based method with the R-method which starts proving very accurate at frequencies where the former starts deteriorating.

Acknowledgements

The authors express their warmest thanks to J-J. Brioist and the whole team in charge of developing CESAR-LCPC for their help during the implementation of all the methods.

REFERENCES

[Bal96] Balmès E. (1996) De l'utilisation de la norme en énergie pour la création de modèles réduits en dynamique des structures. *C. R. Acad. Sci., Paris, série 2* .

[Bd92a] Bourquin F. and d'Hennezel F. (1992) Intrinsic component mode synthesis and plate vibrations. *Comp. and Str.* 44(1): 315–324.

[Bd92b] Bourquin F. and d'Hennezel F. (1992) Numerical study of an intrinsic component mode synthesis method. *Comp. Meth. Appl. Mech. Eng.* 97: 49–76.

[BGLT88] Bourgat J.-F., Glowinski R., and Le Tallec P. (1988) Formulation variationnelle et algorithme de décomposition de domaines pour les problèmes elliptiques. *C. R. Acad. Sci., Paris, série 1* 306: 569–572.

[BKP95] Bramble J., Knyazev V., and Pasciak J. (1995) A subspace preconditioning algorithm for eigenvector/eigenvalue computation. Technical report, University of Colorado at Denver, Center for Computational Mathematics.

[BN97a] Bourquin F. and Namar R. (1997) Extended neumann-neumann preconditioners in view of component mode synthesisin preparation.

[BN97b] Bourquin F. and Namar R. (1997) Extended neumann-neumann preconditioners in view of modal synthesis for platesin preparation.

[Bou89] Bourquin F. (1989) Synthèse modale d'opérateurs elliptiques du second ordre. *C. R. Acad. Sci., Paris, série 1* 309: 919–922.

[Bou90] Bourquin F. (1990) Analysis and comparison of several component mode synthesis methods on one-dimensional domains. *Numer. Math.* 58: 11–34.

[Bou91] Bourquin F. (1991) *Synthèse modale et analyse numérique des multistructures élastiques.* PhD thesis, Université P. et M. Curie, Paris, France.

[Bou92] Bourquin F. (1992) Component mode synthesis and eigenvalues of second order operators: discretization and algorithm. *RAIRO Modélisation Mathématique* *et Analyse Numérique* 26(3): 385–423.

[BVPA94] Babuska I., Von Petersdorff T., and Andersson B. (1994) Numerical treatment of vertex singularities and intensity factors for mixed boundary value problems for the laplace equation in r3. *SIAM J. Numer. Anal.* 31(5): 1265–1288.

[CB68] Craig R. and Bampton M. (1968) Coupling of substructures for dynamic analysis. *A.I.A.A. Jour.* 6: 1313–1321.

[CDVM95] Charpentier I., De Vuyst F., and Maday Y. (1995) A component mode synthesis method of infinite order of accuracy using subdomain overlapping. In *Proceedings of ENUMATH, Paris.*

[CDVM96] Charpentier I., De Vuyst F., and Maday Y. (1996) Méthode de synthèse modale avec une décomposition de domaine par recouvrement. *C. R. Acad. Sci., Paris, série 1* 322: 881–888.

[Cha83] Chatelin F. (1983) *Spectral approximation of linear operators.* Academic Press.

[CL96] Cros J.-M. and Lene F. (1996) Parallel iterative methods to solve large-scale eigenvalue problems in structural dynamics. In *Proc. Ninth Int. Conf. on Domain Decomposition Meths.*

[Clo93] Clouteau (1993) PhD thesis, Ecole Centrale Paris.

[Cra85] Craig R. J. (June 1985) A review of time domain and frequency domain component mode synthesis methods. In *Proceedings of the 85 joint ASCE/ASME mechanics conference*, volume 67. Albuquerque.

[Des89] Destuynder P. (1989) Remarks on dynamic substructuring. *Eur. J. Mech., A/Solids* 8(3): 201–218.

[DO96] Destuynder P. and Ousset Y. (1996) Une méthode de branch mode en sous-structuration dynamique. *C.R. Acad. Sci., Paris, série 1* 322: 91–96.

[dV96] der Vorst V. (1996) A parallelizable and fast algorithm for very large generalized eigenproblems. Technical report, Utrecht University, the Netherlands.

[FG94] Farhat C. and Géradin M. (1994) On a component mode synthesis method and its application to incompatible substructures. *Comp. and Str.* 51(5): 459–473.

[Gib88] Gibert R. J. (1988) *Vibrations des Structures, Interactions avec les fluides, sources d'excitations aléatoires.* Eyrolles. Ecole d'été d'analyse numérique CEA INRIA EDF, 1986.

[Hur65] Hurty W. (1965) Dynamic analysis of structural systems using component modes. *AIAA Jour.* 4(4): 678–685.

[Imb79] Imbert J. (1979) *Calcul des Structures par éléments finis.* Cépadues.

[J́85] Jézéquel L. (1985) *Synthèse modale: théorie et extensions.* PhD thesis, Université Claude Bernard, Lyon, France.

[LM68] Lions J. and Magenes E. (1968) *Problèmes aux limites et Applications*, volume 1. DUNOD, Paris.

[Lui96] Lui S.-H. (1996) Some recent results on domain decomposition methods for eigenvalue problems. In *Proc. Ninth Int. Conf. on Domain Decomposition Meths.*

[Mal92] Maliassov S. (1992) On the analog of the schwarz method for spectral problems. *Numerical Methods and Mathematical Modeling, Inst. Numer. Math., Russian Acad. Sci., Moscow* pages 71–79. in Russian.

[Mal96] Maliassov S. (1996) On the schwarz alternating method for eigenvalue problems. Technical report, Institute for Mathematics and its Applications, Minneapolis, USA.

[Mas88] Masson J.-C. (1988) Présentation générale des méthodes de synthèse modale. Handed out at the Institut pour la promotion des sciences de l'ingénieur, Paris, France.

[Mei80] Meirovitch L. (1980) *Computational Methods in Structural Dynamics.* Sijthoff and Noordhoff.

[MN71] Mac Neal R. (1971) A hybrid method of component mode synthesis. *Comp. and Str* 1(4).

[MO79] Morand H. and Ohayon R. (1979) Substructure variational analysis of the vibrations of coupled fluid-structure systems. finite element results. *Int. Jour. for Num. Meth. in Eng.* 14: 741–755.

[RTG96] Rixen D., Thonon C., and Géradin M. (1996) Impedance and admittance of continuous systems and comparison between continuous and discrete models. In *ESA International Workshop on advanced mathematical methods in the dynamics of flexible bodies, ESTEC.*

[SC96] Sharapov A. and Chan T. (1996) Domain decomposition and multilevel methods for eigenvalue problems. In *Proc. Ninth Int. Conf. on Domain Decomposition Meths.*

[Tra92a] Tran D.-M. (June 1992) Hybrid methods of component mode synthesis using attachment modes or residual attachment modes. In *Proceedings of the 2nd ESA International Workshop on Modal representation of flexible structures by continuum methods, ESTEC.* Noordwijk(The Netherlands).

[Tra92b] Tran D.-M. (1992) Méthodes de synthèse modale mixtes. *Revue Européenne des Eléments Finis* 1(2): 137–179.

[Tra96] Tran D.-M. (1996) Méthode de sous-structuration pour l'analyse de sensibilité et la réactualisation des modes propres des structures localement perturbées. *Revue Européenne des Eléments Finis* to appear.

[Val82] Valid R. (1982) Une méthode de calcul des structures au flambage par sous-structuration et synthèse modale. *C.R. Acad. Sci., série 2* 294: 299–302.

35

Additive Schwarz for the Schur Complement Method

Luiz M. Carvalho and Luc Giraud

1 Introduction

Domain decomposition methods for solving elliptic boundary problems have been receiving increasing attention for the last two decades. To a large extent this is due to their potential application on new parallel computers.

We describe here two domain decomposition algorithms based on nonoverlapping subregions for solving self-adjoint elliptic problems in two dimensions and we report on some experimental results. Both alternatives can be viewed as block diagonal preconditioners for the Schur complement matrix. The first is the classical block Jacobi where all but one of the diagonal blocks are related to the interfaces, the remaining block is diagonal and corresponds to the cross-points. The second preconditioner introduces an overlap between the blocks by including the cross-points and their couplings in the diagonal blocks of the block Jacobi preconditioner. In this case, the block related to the cross-points is dropped. We will refer to the latter as Algebraic Additive Schwarz (AAS) since we sum the contributions of each block on the overlap.

The nonzero sub-blocks of the Schur complement are dense matrices, we consider approximations which are inexpensive to construct and invert. The algebraic approximation of the interface operator is constructed by using the probing technique studied in [CM92] for the two-subdomain case and extended in [CMS92] for many subdomains. The proposed preconditioner belongs to the one-level type preconditioners as, for instance, Dirichlet-Neumann [BW86] and Neumann-Neumann [RT91]. Therefore, it does not implement any coarse grid component. Currently, most preconditioners include a coarse correction to propagate the error globally. We refer the reader to [BPS86, Smi90, Man93] for a detailed presentation of some of these preconditioners and to [SrG96, CM94] for complete surveys of domain decomposition methods. We should stress that the AAS approach, which first appeared in a few numerical experiments in [GT93], improves the convergence rate of the well-known block Jacobi on some model problems while retaining the same computational complexity.

First, in Section 2, we describe the AAS preconditioner. In Section 3, we present

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

the parallel implementation of both preconditioners on distributed memory platforms, and compare their performance in Section 4. We conclude in Section 5.

2 The AAS Preconditioner

We consider the following 2^{nd} order self-adjoint elliptic problem on an open polygonal domain Ω included in $I\!R^2$:

$$\left\{ \begin{array}{rcll} -\frac{\partial}{\partial x}(a(x,y)\frac{\partial u}{\partial x}) - \frac{\partial}{\partial y}(b(x,y)\frac{\partial u}{\partial y}) & = & f(x,y) & \text{in } \Omega \\ u & = & 0 & \text{on } \partial\Omega, \end{array} \right. \tag{2.1}$$

where $a(x,y)$, $b(x,y) \in I\!R^2$ are positive functions on Ω. We assume that the domain Ω is partitioned into N nonoverlapping subdomains $\Omega_1, \dots, \Omega_N$ with boundaries $\partial\Omega_1, \dots, \partial\Omega_N$. We discretise (2.1) by either finite differences or finite elements resulting in a symmetric and positive definite linear system $Au = f$. Grouping the points corresponding to the interfaces between the subdomains (B) in the vector u_B and the ones corresponding to the interior (I) of the subdomains in u_I, we get the reordered problem:

$$\begin{pmatrix} A_{II} & A_{IB} \\ A_{IB}^T & A_{BB} \end{pmatrix} \begin{pmatrix} u_I \\ u_B \end{pmatrix} = \begin{pmatrix} f_I \\ f_B \end{pmatrix} . \tag{2.2}$$

Eliminating u_I from the second block row of (2.2) leads to the following reduced equation for u_B:

$$Su_B = f_B - A_{IB}^T A_{II}^{-1} f_I \ , \qquad \text{where} \qquad S = A_{BB} - A_{IB}^T A_{II}^{-1} A_{IB} \tag{2.3}$$

is referred to as the Schur complement matrix.

Let

$$B = (\bigcup_{i=1}^{m} E_i) \cup V, \tag{2.4}$$

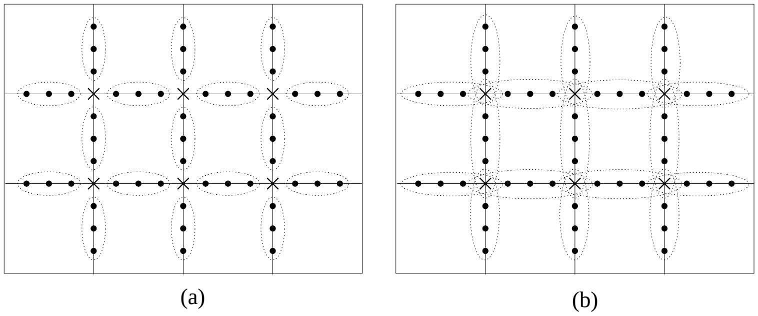
be a partition of the interface B into edge points, E_i, (depicted with $\bullet$ in Figure 1) and vertex points, V, ($\times$ points in Figure 1); E_i can be written as $E_i = (\partial\Omega_j \cap \partial\Omega_l) - V$, with $j \neq l$. Let R_{E_i} and R_V denote the pointwise restriction maps from B onto the nodes on E_i and on V, and let $R_{E_i}^T$ and R_V^T be the corresponding extension maps, respectively. The diagonal block associated with E_i is defined by

$$S_{ii} = R_{E_i} S R_{E_i}^T, \ \ i = 1, \dots, m. \tag{2.5}$$

Let $\tilde{S}_{ii}$ be an approximation for S_{ii}, $\ i = 1, \dots, m$. Then, the approximate block Jacobi preconditioner, bJ, can be represented by

$$bJ = \sum_{i=1}^{m} R_{E_i}^T \tilde{S}_{ii}^{-1} R_{E_i} + R_V^T \tilde{S}_{VV}^{-1} R_V. \tag{2.6}$$

Figure 1 Decomposition of the interface points (a) the block Jacobi and (b) the AAS.



Let

$$B = \bigcup_{i=1}^{m} \hat{E}_i \tag{2.7}$$

be another decomposition of B, where $\hat{E}_i = \partial\Omega_j \cap \partial\Omega_l$ (see Figure 1 (b)). $\hat{E}_i$ and $\hat{E}_j$ may overlap on one point belonging to V, so $\hat{E}_i \cap \hat{E}_j \neq \emptyset$ for some $i \neq j$. Let $\hat{R}_{E_i}$ denote the pointwise restriction map from B onto the nodes on $\hat{E}_i$ and $\hat{R}_{E_i}^T$ the corresponding extension map. We define

$$\breve{S}_{ii} = \hat{R}_{E_i} S \hat{R}_{E_i}^T, \;\; i = 1, \ldots, m. \tag{2.8}$$

Let $\hat{S}_{ii}$ be an approximation for $\breve{S}_{ii}$, $\;i = 1, \ldots, m$. Then, we consider the Algebraic Additive Schwarz (AAS) preconditioner defined as:

$$AAS = \sum_{i=1}^{m} \hat{R}_{E_i}^T \hat{S}_{ii}^{-1} \hat{R}_{E_i} \tag{2.9}$$

3 Parallel Implementation

We use the probe technique to construct $\tilde{S}_{ii}$ and $\hat{S}_{ii}$ [CM92]. Let $P = [p_i]$ be a matrix whose columns are the probe vectors p_i. Therefore, the probe technique can be applied by multiplying S with P. This matrix-matrix approach only requires two communications for the construction and is more efficient than a more classical approach based on a sequence of matrix-probing vector products. Moreover, the matrix-matrix approach allows the algorithm to overlap part of the communication with some computation.

The implementation of $\hat{R}_{E_i}^T \hat{S}_{ii}^{-1} \hat{R}_{E_i}$ in AAS requires two extra neighbour-neighbour communications to exchange information on the cross-points; these communications are not needed for block Jacobi.

The classical PCG requires two synchronisations to compute the inner products; we have implemented the variant proposed in [DER93] where only one synchronisation per iteration is needed.

4 Experimental Results

The elliptic problem (2.1) was discretised by the standard five-point difference stencil on an $(n \times n)$ mesh. The approximations $\tilde{S}_{ii}$ and $\hat{S}_{ii}$ are tridiagonal matrices built using twelve probing vectors [CG97]. The direct solver for the solution of the local Dirichlet problems A_{ii}, $i = 1 \ldots N$, is MA27 [DR83], while the tridiagonal matrices $\tilde{S}_{ii}$ and $\hat{S}_{ii}$ are factorised using LAPACK routines. The stopping criterion is to decrease the relative residual to less than 10^{-5}. The grid is either uniform or nonuniform. The message passing library used is MPI. The parallel platform is a Cray T3D (IDRIS - France). In the parallel experiments, we have used as many processors as subdomains and the computations have been performed in 64 bit precision.

Figure 2 Definition of the function $a(.,.)$ on Ω, the unit square of $\mathbb{R}^2$.

a(.,.) = 10^{-3}	a(.,.) = 10^{3}	a(.,.) = 10^{-3}
a(.,.) = 10^{3}	a(.,.) = 10^{-3}	a(.,.) = 10^{3}

Table 1 gives the number of iterations of the PCG. Table 2 gives the total parallel elapsed time, in seconds, for solving $Ax = b$; that is, the analysis and factorisation steps of MA27, the preconditioner construction, the PCG iterations, and the computation of the global solution by local back substitutions.

Table 1 Number of iterations of the PCG on a 257 × 257 grid.

	AAS			block Jacobi		
# subdomains	Poisson	Aniso.	Nonunif.	Poisson	Aniso.	Nonunif.
4	17	11	14	18	11	16
16	23	29	27	24	35	37
64	33	52	40	35	82	60

Table 1 shows that AAS performs better than block Jacobi in the presence of anisotropic phenomena; the increase in the number of iterations is less for AAS as the number of subdomains grows larger. We have observed such behaviour for a wide class of model test problems, but we only report two of these results here. For all the experiments the function $b(.,.)$ has been set to one. In the first example, the anisotropy is physically present in the definition of the problem through the function

Table 2 Times (in sec.) on the Cray T3D for solving $Ax = b$ on a 257×257 grid.

	AAS			block Jacobi		
# subdomains	Poisson	Aniso.	Nonunif.	Poisson	Aniso.	Nonunif.
4	10.77	9.64	10.18	10.97	9.64	10.57
16	2.17	2.43	2.33	2.20	2.66	2.72
64	0.58	0.78	0.65	0.59	1.07	0.84

$a(.,.)$ as defined in Figure 2, while in the second example the anisotropy is numerically introduced in the Poisson problem by the discretisation on a nonuniform grid. This grid is used to solve the drift-diffusion equations [GT93] involved in MOSFET device simulation.

As a first remark, we indicate that the time per iteration is almost the same for PCG with both preconditioners; the neighbour-neighbour communications required by AAS do not penalise its performance on the Cray T3D.

We can observe in Table 2 that the reduction in the number of PCG iterations is not directly reflected in the time reduction. This is due to the significant amount of time required by the MA27 analysis and the factorisation routines for the solution of the local problems A_{ii}, $i = 1 \ldots N$, which are common to both preconditioners. These routines are responsible for 20% of the global solution time in the case of 64 subdomains. Consequently, the global time is not a linear function of the number of iterations but is an affine function; that is, the global parallel solution time is equal to the time for the symbolic and numerical factorisation of the local Dirichlet matrices, plus the time for the back solution (once the interface problem has been solved), plus the number of iterations times the time per iteration (which is the same for both preconditioners).

Table 3 shows the performance of the parallel device simulation code, using PCG with either AAS or block Jacobi for the solution of the linear systems involved in the nonlinear solver. Both numerical and physical anisotropies are encountered and AAS performs better than block Jacobi.

Table 3 Times in seconds and number of iterations for a continuation step of the MOSFET device simulation on a 257×257 grid.

	AAS		block Jacobi	
# subdomains	time	# iterations	time	# iterations
4	857.6	1520	861.6	1543
16	214.3	2731	240.1	3422
64	67.5	4877	88.9	7300

The AAS preconditioner can easily be extended by using a larger overlap between the edges E_i. Experimental results have shown that this does not significantly accelerate the convergence for the MOSFET problem. In Table 4, we vary the size of the overlap and report the total number of linear iterations involved in the solution of the drift-diffusion equations.

Table 4 Iterations using preconditioners with overlap sizes 0 (block Jacobi), 1(AAS), 2 and 3 on a 129×129 grid for the MOSFET problem.

	# points in the overlap between the $\hat{E}_i$			
# subdomains	0	1	3	5
4	1660	1693	1678	1671
16	3610	2842	2797	2720
64	8385	5177	5419	5599

5 Conclusions

We have compared a so-called Algebraic Additive Schwarz preconditioner (AAS) with an approximate block Jacobi preconditioner for the Schur complement domain decomposition method for solving elliptic PDEs. We have shown that AAS performs better both in terms of number of iterations and in computing time, for problems with anisotropic phenomena. This behaviour has been illustrated when solving linear systems that arise from model problems and from the real life application of device modelling.

Although AAS does not introduce a global coupling among the subdomains, the growth of the number of iterations is slower than the one observed with block Jacobi, when the number of subdomains increases. AAS can be a cheap alternative to improve the simple block Jacobi preconditioner when it is difficult to define or to implement a coarse problem in the preconditioner applied to the Schur complement system.

For a detailed presentation of this work we refer to [CG97].

Acknowledgement

We would like to thank the referee whose comments helped to improve the presentation of this paper.

REFERENCES

[BPS86] Bramble J., Pasciak J., and Schatz J. (1986) The construction of preconditioners for elliptic problems by substructuring I. *Math. Comp.* 47(175): 103–134.

[BW86] Bjørstad P. E. and Widlund O. B. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM Journal on*

Numerical Analysis 23(6): 1093–1120.

[CG97] Carvalho L. M. and Giraud L. (1997) Parallel domain decomposition for device modelling (in preparation). Technical report, CERFACS, Toulouse, France.

[CM92] Chan T. F. and Mathew T. P. (1992) The interface probing technique in domain decomposition. *SIAM J. Matrix Analysis and Applications* 13.

[CM94] Chan T. F. and Mathew T. P. (1994) *Domain Decomposition Algorithms*, volume 3 of *Acta Numerica*, pages 61–143. Cambridge University Press, Cambridge.

[CMS92] Chan T., Mathew T., and Shao J.-P. (1992) Fourier and probe variants of the vertex space domain decomposition algorithm. In Keyes D., Chan T., Meurant G., Scroggs J., and Voigt R. (eds) *Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 236–249. SIAM, Phil.

[DER93] D'Azevedo E., Eijkhout V., and Romine C. (1993) Reducing communication costs in the conjugate gradient algorithm on distributed memory multiprocessors. Technical Report CS-93-185, Computer Science Department, University of Tennessee, Knoxville.

[DR83] Duff I. S. and Reid J. K. (1983) The multifrontal solution of indefinite sparse symmetric linear systems. *TOMS* 9: 302–325.

[GT93] Giraud L. and Tuminaro R. (1993) Domain decomposition algorithms for the drift-diffusion equations. In Sincovec R., Keyes D., Leuze M., Petzold L., and Reed D. (eds) *Proceedings of the Sixth SIAM Conference on Parallel Processing for Scientific Computing*, pages 719–726. SIAM.

[Man93] Mandel J. (1993) Balancing domain decomposition. *Comm. Numer. Meth. Engrg.* 9: 233–241.

[RT91] Roeck Y.-H. D. and Tallec P. L. (1991) Analysis and test of a local domain decomposition preconditioner. In Glowinski R., Kuznetsov Y., Meurant G., Périaux J., and Widlund O. (eds) *Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations.* SIAM, Philadelphia, PA, USA.

[Smi90] Smith B. F. (1990) *Domain decomposition algorithms for the partial differential equations of linear elasticity.* PhD thesis, Courant Institute of Mathematical Sciences, New York.

[SrG96] Smith B., rstad P. B., and Gropp W. (1996) *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, New York, 1st edition.

36

Subspace Correction Multilevel Methods for Elliptic Eigenvalue Problems

Tony F. Chan and Ilya Sharapov

1 Introduction

Domain decomposition and multigrid methods are powerful techniques for solving elliptic linear problems. Unfortunately the straightforward implementation of the methods is limited to linear problems and relatively little work has been done for nonlinear applications. The goal of this paper is to analyze the application of the multiplicative Schwarz methods to the eigenvalue problem without linearization. An important distinction of this approach is that the subspace problem is also a generalized eigenvalue problem which allows to apply the algorithm recursively and formulate a multilevel method of optimal complexity.

Solution of eigenvalue problems by multigrid methods using linearization was discussed by Hackbusch ([Hac84]) and McCormick ([McC92]). The idea to use coordinate relaxation applied directly for a matrix eigenvalue problem goes back to the book by Fadeev and Fadeeva [FF63] (1963) where they applied a technique similar to Gauss-Seidel method for minimizing the Rayleigh quotient. This approach was extended by Kaschiev [Kas88] and Maliassov [Mal92] for PDE-based problems. In this case the resulting method of minimizing the Rayleigh quotient is analogous to the block Gauss-Seidel method for linear problems.

Several domain decomposition-based methods were proposed by Lui [Lui95], in particular the method based on a nonoverlapping partitioning where the interface problem is solved either using a discrete analogue of a Steklov-Poincare operator or using Schur complement-based techniques. The former approach resembles the component mode synthesis method (cf. Bourquin and Hennezel [BH92]) which is an approximation rather than iterative technique for solving eigenproblems. The component mode synthesis was also used by Farhat and Géradin in [FG94]. Stathopoulos, Saad and Fischer [SSF95] considered iterations based on the Schur complement of the block corresponding to the interface variables.

Anther way to implement the domain decomposition technique on eigenvalue

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

problems is the divide and conquer method proposed by Dongarra and Sorensen [DS87]. An attempt to link relaxation methods (in particular SOR) to eigenvalue problem was made by Ruhe [Ruh74].

In this work we extend the results of [Mal92] and [Kas88] for the multiplicative Schwarz method by considering the two-level scheme. Convergence is proved for a more suitable class of initial approximations and an asymptotic convergence analysis is given. We also describe the recursive implementation of the method, which results in a multilevel algorithm. Finally we present an alternative variational formulation of the problem, which is equivalent mathematically but more suitable for theoretical considerations.

2 Subspace Correction for Eigenvalue Problems

Let us consider the problem of finding the minimal eigenvalue λ and the corresponding eigenvector u of

$$Lu \equiv -\sum_{i,j=1}^{2} \frac{\partial}{\partial x_i} a_{i,j} \frac{\partial u}{\partial x_j} + p(x)u = \lambda u \tag{2.1}$$

$$x \in \Omega, \qquad u \mid_{\partial\Omega} = 0 \qquad a_{i,j} > 0 \ ,$$

where Ω is a bounded region in R^2 and $a_{i,j}(x) = a_{j,i}(x)$, $p(x) \geq 0$ are piecewise smooth real functions.

To discretize the problem, we can perform a triangulation of Ω with triangles of quasi-uniform size h and use the standard finite element approach to represent (2.1) as

$$Au = \lambda Mu \ , \tag{2.2}$$

where $A = A^T > 0$ and $M = M^T > 0$ are stiffness and mass matrices respectively. The problem of finding the minimal eigenvalue of (2.2) can be viewed as a minimization of the Rayleigh quotient

$$\lambda_1 = \min_u F(u) = \min_u \frac{u^T Au}{u^T Mu} \ . \tag{2.3}$$

In order to apply domain decomposition technique to this problem we can represent Ω as a union of overlapping subdomains with Lipschitz boundaries: $\Omega = \cup_{i=1}^{J} \Omega_i$. Let $\{V_i\}_{i=1}^{J}$ be finite element subspaces corresponding to this partition and let P_i^T denote the orthogonal projection into the subspace V_i; its transpose P_i is the prolongation operator from V_i to H_h^2. We also introduce the M-norm of a vector $\|v\|_M = (v^T Mv)^{1/2}$.

A scheme analogous to the multiplicative Schwarz algorithm for solving (2.2) was proposed in [Mal92] and [Kas88]:

```
Algorithm 1 (Multiplicative subspace correction)
    Starting with $u^0$ for $k = 0$ until convergence
        for $i = 1 : J$
            find $u^{k+i/J}$ such that
                $F(u^{k+i/J}) = \min_{d_i \in V_i} F(u^{k+(i-1)/J} + P_i d_i)$        (2.4)
        end
    end
```

To have some control over the norm of the iterates we can M-normalize the approximations either after each subiteration or after a loop over all subdomains is completed.

At each step the algorithm performs a subspace search minimizing the Rayleigh quotient using the correction from the current subspace. We will now show that the minimization (2.4) results in minimizing the Rayleigh quotient for the local $(n_i + 1) \times (n_i + 1)$ problem, where n_i is the dimension of the current subspace.

Rewrite (2.4) as

$$\rho(u^{k+i/J}) = \min_{\tilde{d}_i} \rho(\tilde{P}_i \tilde{d}_i) = \min_{\tilde{d}_i} \frac{(\tilde{P}_i\tilde{d}_i)^T A (\tilde{P}_i\tilde{d}_i)}{(\tilde{P}_i\tilde{d}_i)^T M (\tilde{P}_i\tilde{d}_i)} = \min_{\tilde{d}_i} \frac{\tilde{d}_i^T \tilde{A} \tilde{d}_i}{\tilde{d}_i^T \tilde{M} \tilde{d}_i} \ ,$$

where

$$\tilde{d}_i = \begin{pmatrix} d_i \\ 1 \end{pmatrix}, \qquad \tilde{P}_i = \begin{pmatrix} P_i & u^{k+(i-1)/J} \end{pmatrix}$$

and

$$\tilde{A} = \tilde{P}_i^T A \tilde{P}_i, \qquad \tilde{M} = \tilde{P}_i^T M \tilde{P}_i \ . \tag{2.5}$$

Thus the subspace problem is an eigenvalue problem with $\tilde{A}$ and $\tilde{M}$.

The matrices $\tilde{A}$ and $\tilde{M}$ preserve the sparsity of the original matrices A and M except for the last row and column, therefore the minimization subproblem can be efficiently solved.

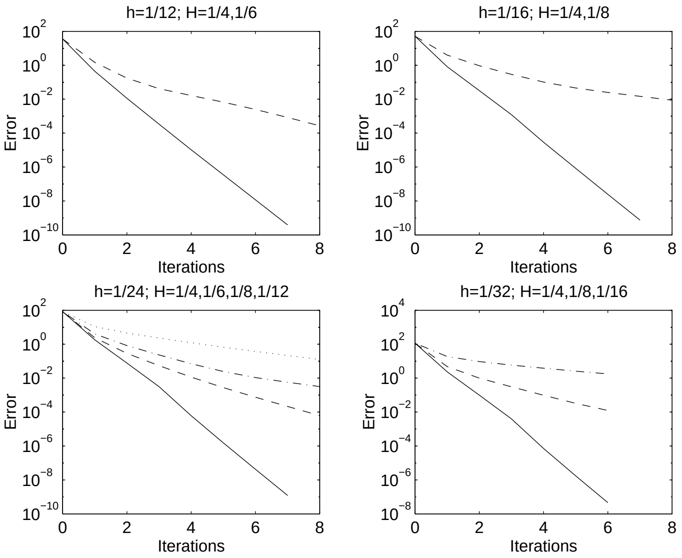
Convergence for **Algorithm 1** was proven in [Mal92] and [Kas88] with the assumption that the initial guess u^0 satisfies

$$\lambda_1 < F(u^0) < \lambda_2 \ . \tag{2.6}$$

Lui [Lui96] pointed out that the algorithm can break down in certain degenerate cases and proposed a modified procedure proving convergence for the case of two subdomains under condition (2.6).

This condition is difficult to control unless we use the method as some refinement procedure using it after a good approximation to the lowest eigenmode was produced by some other method. We can formulate a stronger result that **Algorithm 1** converges to the lowest eigenpair of (2.2) in the case of any number of subdomains and with the more practical assumption that all the components of the initial approximation u^0 are of the same sign.

Figure 1 Error reduction for the model problem without the coarse grid correction. Higher curves show slow convergence for the large number of subdomains.



Theorem 1 *Vectors u^k and the corresponding Rayleigh quotients $\rho(u^k)$ produced by* **Algorithm 1** *converge to the lowest eigenmode of discretized problem (2.2) if all the components of the initial approximation u^0 satisfy $u_i^0 > 0$.*

The proof (to appear in a full version of this paper) relies on a natural assumption that the eigenvector we are looking for is not contained in any of the subspaces V_i:

Assumption 1 *For any subspace V_i there is a constant $C_i > \lambda_1$ such that*

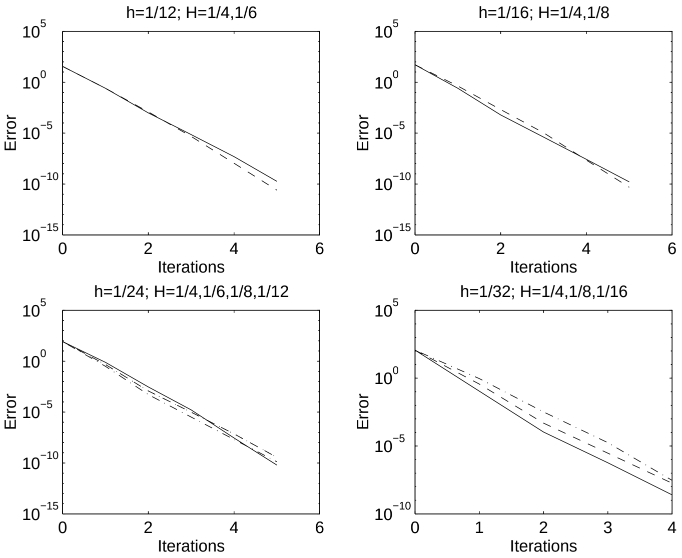
$$v^T A v \geq C_i v^T M v \qquad \text{forany} \qquad v \in V_i \; .$$

3 Coarse Grid Correction and Multilevel Method

In this work we modify **Algorithm 1** by adding a coarse grid correction after a loop over the subdomains is completed. By doing so in the case of a linear elliptic problem with sufficient subdomain overlap, the convergence rate becomes independent of both the meshsize and the number of subdomains [BPWX91, Xu92].

The effect of the coarse grid correction for a model problem of the 2D Laplacian is shown in the following figures, where h and H are fine and coarse meshsizes respectively. We see that without the coarse grid the convergence rate is dependent on both the meshsize and the number of subdomains, whereas after the inclusion of the coarse grid correction the convergence rate becomes independent of both h and H.

Figure 2 Error reduction for the the model problem with the coarse grid correction. Convergence is independent of the number of subdomains.



Since the subspace problem is of the same type as the original one, i.e. a generalized eigenvalue problem, we can make the algorithm more efficient by applying it recursively. Instead of solving the eigenvalue subproblem over a subdomain by some other method we can apply several iterations of the same algorithm. Applying that recursion to the multiplicative method with coarse grid correction we can view the resulting scheme as a multilevel method and the iterations performed on each level as the smoothing of the solution (in the spirit of the multigrid method). The recursion can be stopped once the subproblems reach some sufficiently small fixed size.

Though the algorithms as presented are sequential we can add some degree of parallelism using multicoloring techniques (see, e.g., [CM94]).

4 Alternative Formulation

A different variational formulation for the symmetric positive definite eigenvalue problem (2.2) was recently proposed by Mathew and Reddy (94) [MR94]. They pointed out that the minimal eigenpair (u_1, λ_1) can be characterized as:

$$J(u_1) = \min_v J(v) \equiv \min_v \left[v^T A v + \mu(1 - v^T M v)^2\right] \tag{4.7}$$

with

$$\lambda_1 = 2\mu - \sqrt{4\mu^2 - 4\mu J(u_1)}$$

and

$$\|u_1\|_M^2 = 1 - \frac{\lambda_1}{2\mu}$$

for any

$$\mu > \lambda_1/2\ . \tag{4.8}$$

Unlike the Rayleigh quotient minimization, formulation (4.7) is unconstrained. The μ-term in $J(v)$ serves as a barrier to pull the solution u away from the trivial solution.

The subspace problem for (4.7) is again of the same form as the original one with dimension $n_i + 1$, i.e. an eigenvalue problem of this size. For

$$\tilde{P}_i = \begin{pmatrix} P_i & u^{k+(i-1)/J} \end{pmatrix}, \qquad \tilde{d}_i = \begin{pmatrix} d_i \\ \alpha \end{pmatrix},$$

we can write the minimization step of the algorithm as

$$J(u^{k+i/J}) = \min_{d_i \in V_i, \alpha} J(\tilde{P}_i \tilde{d}_i)$$

$$= \min_{\tilde{d}_i} \left[(\tilde{P}_i\tilde{d}_i)^T A (\tilde{P}_i\tilde{d}_i) + \mu (1 - (\tilde{P}_i\tilde{d}_i)^T M (\tilde{P}_i\tilde{d}_i))^2 \right]$$

$$= \min_{\tilde{d}_i} \left[\tilde{d}_i^{\,T} \tilde{A} \tilde{d}_i + \mu (1 - \tilde{d}_i^{\,T} \tilde{M} \tilde{d}_i)^2 \right] \ ,$$

where

$$\tilde{A} = \tilde{P}_i^{\,T} A \tilde{P}_i, \qquad \tilde{M} = \tilde{P}_i^{\,T} M \tilde{P}_i$$

Therefore, we can see that for any choice of μ satisfying (4.8), one subspace correction step for formulations (2.3) and (4.7) results in the same reduced generalized eigenvalue problem with matrices (2.5). The application of the multiplicative Schwarz algorithm to both formulations results in the same approximations to the lowest eigenvalue and the approximations to the eigenvector are the same up to scaling.

The objective function in (4.7) is convex near the solution so for the local analysis we can use apply the theory of multiplicative Schwarz methods for minimization problems [TE96]. Equivalence of the formulations gives the asymptotic result:

Theorem 2 *The iterates produced by* **Algorithm 1** *with coarse grid correction applied to formulations (2.3) and (4.7) (in SPD case) satisfy for k large enough*

$$\|u_1 - u^{k+1}\| \le (1-\delta)\|u_1 - u^k\| \ ,$$

where (u_1, λ_1) *is the minimal eigenpair of (2.2) with* $\|u_1\|_M = 1$ *for (2.3) and* $\|u_1\|_M = 1 - \frac{\lambda_1}{2\mu}$ *for (4.7) and the value of* $\delta > 0$ *is independent of the meshsize h and the number of subdomains J.*

The proof of this theorem will be given in a larger version of this paper.

Acknowledgement

The authors wish to thank Jinchao Xu and Xue-Cheng Tai for their helpful suggestions during discussions on the topic of this paper. This work was partially supported by the ARO under contract DAAL-03-9-C-0047 (Univ. of Tenn. subcontract ORA 4466.04, Amendment 1 and 2) and ONR under contract ONR-N00014-92-J-1890.

REFERENCES

[BH92] Bourquin F. and Hennezel F. (1992) Application of domain decomposition techniques to modal synthesis for eigenvalue problems. In *Fifth International Symposium on Domain Decomposition Methods for Partial Differential Equations (Norfolk, VA, 1991)*, pages 214–223. SIAM, Philadelphia.

[BPWX91] Bramble J., Pasciak J., Wang J., and Xu J. (1991) Convergence estimates for product iterative methods with applications to domain decomposition. *Math. Comp.* 57(195): 1–21.

[CM94] Chan T. and Mathew T. (1994) Domain decomposition algorithms. In *Acta Numerica*, pages 61–143. Cambridge Univ. Press, Cambridge.

[DS87] Dongarra J. and Sorensen D. (1987) A fully parallel algorithm for symmetric eigenvalue problem. *SIAM J. Sci. Statist. Comput.* 8(2): 139–154.

[FF63] Fadeev D. and Fadeeva V. (1963) *Computational Methods of Linear Algebra.* W.H. Freeman and Company, San Francisco.

[FG94] Farhat C. and Géradin M. (1994) On a component mode synthesis method and its application to incompatible substructures. *Computers and Structures* 51: 459–473.

[Hac84] Hackbusch W. (1984) *Multigrid Methods.* Springer-Verlag.

[Kas88] Kaschiev M. (1988) An iterative method for minimization of the Rayleigh-Ritz functional. In *Computational processes and systemss, No. 6 (Russian)*, pages 160–170. Nauka, Moscow.

[Lui95] Lui S. (1995) Domain decomposition for eigenvalue problems (preprint). Hong Kong Univ. of Science and Tech.

[Lui96] Lui S. (1996) On two Schwarz alternating methods for the symmetric eigenvalue problem (preprint). Hong Kong Univ. of Science and Tech.

[Mal92] Maliassov S. (1992) On the analog of Schwarz method for spectral problems. In *Numerical methods and mathematical modeling (Russian)*, pages 70–79. Otdel Vychisl. Mat., Moscow.

[McC92] McCormick S. (1992) *Multilevel Projection Methods for Partial Differential Equations.* SIAM, Philadelphia.

[MR94] Mathew G. and Reddy V. (1994) Development and analysis of a neural network approach to Pisarenko's harmonic retrieval method. *IEEE Trans. Sig. Proc.* 42(3): 663 – 673.

[Ruh74] Ruhe A. (1974) SOR-methods for the eigenvalue problem with large sparse matrices. *Math. Comput.* 28: 695–710.

[SSF95] Stathopoulos A., Saad Y., and Fischer C. (April 2-7 1995) A Schur complement method for eigenvalue problems. In *Proceedings of the Seventh Copper Mountain Conference on Multigrid Methods.*

[TE96] Tai X.-C. and Espedal M. (1996) Rate of convergence of a space decomposition method and applications liear and nonlinear ellicptic problems (preprint). Univ. of Bergen.

[Xu92] Xu J. (1992) Iterative methods by space decomposition and subspace correction. *SIAM Rev.* 34: 581–613.

37

Parallel Iterative Methods for Large-scale Eigenvalue Problems in Structural Dynamics

Jean-Michel Cros, Françoise Léné

1 Introduction

Consider the following generalized eigenvalue problem:

$$Kq = \lambda Mq \tag{1.1}$$

where K and M are respectively the symmetric stiffness matrix and mass matrix, the eigenvalues λ are the squares of the natural frequencies ω, and q are the eigenvectors, although only the smallest eigenpairs are wanted. We are interested in problems where K and M are very large (more than 10^4 unknowns), sparse symmetric positive definite (or semi-definite) matrices. The main difficulty is to deal with large matrices which exceed primary memory capacity of sequential computer. Distributed memory architectures have enough memory, and to take advantage of these computers algorithms must involve large tasks that can be executed in parallel. Iterative methods represent a way of developing such algorithms. Iteration techniques to solve problem (1.1), require that the system is reduced to a standard eigenproblem:

$$K^{-1}Mq = \frac{1}{\lambda}q \tag{1.2}$$

and thus a linear system has to be solved (either K^{-1} or an approximation of K^{-1} [SVdV96], [BKP97]). However in structural analysis many problems occur, such as mutiplicity of eigenvalues, semi-definiteness, etc. [GLS86], and some of these have been adressed by the developments in the Lanczos algorithm. In addition the Lanczos method seems to be the most well suited algorithm for very large problems, because it requires few iterations per converged eigenvalue, and this number remains independent of the problem size. But for large problems, a robust and parallel linear solver is required without the use of secondary menory during computation. In static analysis, for geometrical or materials reasons, the stiffness matrix is often ill-conditioned. Iterative substructuring or domain decomposition methods, such as Schur complement

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

methods, have a nice mechanical interpretation. They have proven their numerical and parallel scalability and are better than a direct method [FC95] for this kind of problem. From the CPU time point of view, the Schur dual complement method is more attractive than the primal approach [Cro97], due to the use of an economical preconditioner, and the ease to solve in a parallel way the coarse grid induced by rigid body. Therefore, the large sparse linear system, at each iteration of the Lanczos algorithm, is solved by the dual Schur complement method.

The paper is organized as follows: section 2 recalls briefly the Lanczos method and its parallel implementation. Section 3 presents the way of computing the global rigid-body modes. Section 4, describes a restarting technique to take into account the successive right-hand sides in order to reduce the number of iterations and section 5 is devoted to an extension of this technique. In Section 6, some numerical results obtained on the Intel PARAGON computer, using the finite element package MODULEF (INRIA) in a message-passing environment, are presented.

2 Lanczos Algorithm

The Lanczos algorithm for extracting the smallest eigenpairs of a system is an inverse power-based method. In its basic form, it is an algorithm for computing an orthogonal basis of a Krylov subspace, i.e., a subspace of the form:

$$\mathcal{K}_r = span\{y_0, K^{-1}My_0, ..., (K^{-1}M)^{r-1}y_0\}. \tag{2.3}$$

The main iteration of the algorithm can be briefly described by the following recurrence

$$\beta_r y_{r+1} = K^{-1}My_r - \alpha_r y_r - \beta_{r-1}y_{r-1},$$

where α_r and β_{r-1} are selected in such a way that the vector y_{r+1} is M-orthogonal to y_r and y_{r-1}:

$$\alpha_r = y_r^T MK^{-1}My_r \quad \text{and} \quad \beta_{r-1} = y_r^T MK^{-1}My_{r-1}. \tag{2.4}$$

Then, (2.4) can be expressed in matrix form as follows:

$$K^{-1}MY_r = Y_rT_r + S \;\text{ with } Y_r = [\, y_0 \,...\, y_r \,] \text{ and } S = [\, 0 \,...\, \beta_{r+1}y_{r+1} \,], \tag{2.5}$$

where T_r is a tridiagonal matrix. The application of the Rayleigh-Ritz procedure to the standard form of the initial eigenvalue problem (1.1), by a projection into the subspace generated by the Lanczos vectors $q = Y_r z$, leads to the reduced eigenvalue problem:

$$T_r z = \frac{1}{\omega^2}z. \tag{2.6}$$

We refer to [CG82] for practical considerations of the Lanczos algorithm, such as: choice of starting vector, restart procedure to take into account possible multiple eigenvalues, convergence strategy, eigenmodes and error analysis.

From the numerical point of view the most CPU time consuming operations are:

- solution of the large-scale linear system with K^{-1}
- M-orthogonalization of y_r
- computation of matrix-vector products with M

These tasks are naturally parallelized by substructuring. The physical domain is divided in N_s nonoverlapping subdomains, and the problem on a global domain is replacing by solving iteratively a condensed problem on the interface of the subdomains. This involves at each iteration solving of locally independant problems. Then, each subdomain is allocated to a processor which knows only the data corresponding to its subdomain and information about neighboring subdomains through interface decomposition. As a sequel, a processor is responsible for computing a fixed subset of each vector (Lanczos vector, search directions,etc.). The internal problem in each subdomain is solved by a direct method while the interface problem, which incorporates a coarse grid induced by rigid body modes of subdomains without external Dirichlet conditions, is handled by a parallel Preconditioned Conjugate Projected Gradient (PCPG) method [FR94]. Finally, a full reorthogonalization of the Lanczos vectors is performed. The reduced eigenvalue problem (2.6) is solved in a sequential way thanks to suitable methods from optimized LAPACK library.

3 Global Rigid-body Modes within Substructuring Framework

Structures having rigid-body modes arise frequently, especially for aeronautic applications. In these cases the inverse iteration process which consists in:

$$\begin{aligned} &1) \quad g_r &= My_r \qquad (3.7)\\ &2) \quad y_{r+1} &= K^{-1}g_r \qquad (3.8) \end{aligned}$$

is modified to filter rigid-body modes [GR94] and becomes:

$$\begin{aligned} &1) \quad g_r &= My_r \qquad (3.9)\\ &2) \quad \tilde{g}_r &= g_r - (MR, g_r)R \qquad (3.10)\\ &3) \quad \tilde{y}_{r+1} &= K^{-1}\tilde{g}_r \qquad (3.11)\\ &4) \quad y_{r+1} &= \tilde{y}_{r+1} - (MR, \tilde{y}_{r+1})R \qquad (3.12) \end{aligned}$$

where the matrix R stores the rigid-body modes of the structure. Step 2 and step 4 respectively express the self-equilibrium of the inertia load, and the fact that the new Lanczos vector is M-orthogonalized with respect to the rigid-body modes. The difficulty is then to compute the global rigid-body modes of the structure. We recall the algebraic system induced by the dual Schur complement method [FR94], and we note with an (s) superscript, a matrix or a vector quantity associated to the s^{th} given substructure:

$$K^{(s)}u^{(s)} = f^{(s)} - B^{(s)^T}\mu \quad \text{in } \Omega^{(s)}, \text{ for } s = 1, ..., N_s, \qquad (3.13)$$

$$\sum_{s=1}^{N_s} B^{(s)}u^{(s)} = 0 \quad \text{on } \Gamma, \qquad (3.14)$$

where the vector of Lagrange mutipliers μ represents the interaction forces between the substucture $\Omega^{(s)}$ with $s = 1, ..., N_s$ along their interface Γ, u is the displacement vector, f the loading vector, and $B^{(s)}$ is a signed boolean matrix which localizes a substructure quantity to the substructure interface $\Gamma^{(s)}$.

If the global structure has no Dirichlet boundary conditions, it will be considered as floating. Hence the stiffness matrix is singular and the restriction of its rigid-body mode u_r in each subdomain verifies the following relations:

$$K^{(s)} u_r^{(s)} = 0 \quad \text{in } \Omega^{(s)}, \text{ for } s = 1, ..., N_s, \tag{3.15}$$

$$\sum_{s=1}^{N_s} B^{(s)} u_r^{(s)} = 0 \quad \text{on } \Gamma. \tag{3.16}$$

Let us then introduce the convention of Farhat and Roux [FR94], the following new quantities:

$$G^{(s)} = B^{(s)} R^{(s)} \quad \text{and } G = [G^{(1)} \ldots G^{(N_f)}], \tag{3.17}$$

where the matrix $R^{(s)}$ stores the rigid-body modes of the substructure $\Omega^{(s)}$. In such a situation, the number of floating substructures N_f, is equal to the number of substructures N_s and G does not have full column rank, thus a set of nonzero coefficient Ψ exists such that:

$$\sum_{s=1}^{N_s} B^{(s)} R^{(s)} \Psi^{(s)} = 0 \quad \text{on } \Gamma. \tag{3.18}$$

This equation implies that the rigid displacement field defined by $R^{(s)}\Psi^{(s)}$ in each substructure is continuous across the substructure interface and satisfies definition (3.15), or in other words, the *global* rigid-body modes can be expressed as a linear arrangement of the *local* rigid-body modes. It can be shown [LT90] that $Ker(G) = Ker(G^T G)$, and then Ψ is also solution of the following problem:

$$G^T G \, \Psi = 0 \quad \text{where } \Psi^T = \{\Psi^{(1)^T}, ..., \Psi^{(N_s)^T}\}, \tag{3.19}$$

which provides an easy way to compute the matrix Ψ. To solve problem (3.19), matrix $G^T G$ must be assembled to compute singularities. Let us note that the coarse problem $G^T G$ is the same as previously where the decomposition induces some floating $(N_f < N_s)$ substructures, but the original problem is well posed. Finally, the relations (3.10) and (3.12) are respectively replaced by:

$$\tilde{g}_r^{(s)} = g_r^{(s)} - \gamma \, R^{(s)} \Psi^{(s)} \quad \text{with } \gamma = \sum_{s=1}^{N_s} (M^{(s)} R^{(s)} \Psi^{(s)}, g_r^{(s)}) \tag{3.20}$$

$$\tilde{y}_{r+1}^{(s)} = \tilde{y}_{r+1}^{(s)} - \gamma \, R^{(s)} \Psi^{(s)} \quad \text{with } \gamma = \sum_{s=1}^{N_s} (M^{(s)} R^{(s)} \Psi^{(s)}, \tilde{y}_{r+1}^{(s)}) \tag{3.21}$$

4 Successive Right-hand Sides

The presented technique has been analyzed by Saad [Saa87], and applied to improve substructure based iterative solver for different applications [FC95] [RTD95]. The domain decomposition method leads to an interface problem which is solved thanks to a conjugate gradient method.

$$C\mu = b, \tag{4.22}$$

where C is the interface operator. The conjugate gradient algorithm generates an orthogonal basis for the Krylov subspace $\mathcal{K}_k = span(g_0, Cg_0, ..., C^{k-1}g_0)$, where $g_0 = b - C\mu_0$ is the initial residual, and μ_0 an initial guess. Let us assume known the k first search directions, thus the approximate solution at the $(k+1)^{th}$ iteration can be expressed by:

$$\mu_{k+1} = \mu_0 + \sum_{i=1}^{k} \frac{(b - C\mu_0, w_i)}{(Cw_i, w_i)} w_i. \tag{4.23}$$

At each iteration of the Lanczos algorithm, the same interface problem has to be solved with a new right-hand side:

$$C\,\mu^2 = b^2. \tag{4.24}$$

The information (p^1 search directions collected) from the solution of the first system (4.22) is used to provide an optimal guess $\mu^2_{0,opt}$ for the solution of the second system thanks to expression (4.23).

$$\mu^2_{0,opt} = \mu^2_0 + \sum_{i=1}^{p^1} \frac{(b^2 - C\mu^2_0, w_i)}{(Cw_i, w_i)} w_i. \tag{4.25}$$

The generalization of this restarting procedure for many right-hand sides is given by:

$$\mu^m_{0,opt} = \mu^m_0 + \sum_{i=1}^{p^1+\ldots+p^{m-1}} \frac{(b^m - C\mu^m_0, w_i)}{(Cw_i, w_i)} w_i. \tag{4.26}$$

In practice, the initial guess solution μ^m_0 is chosen equal to zero, which significantly simplifies the computation of the expression (4.26). Let us note that a full reorthogonalization of the search directions is necessary [Rou94] to ensure stability of the algorithm and to avoid computing the same search directions again:

$$w_{k+1} = g_{k+1} - \sum_{i=1}^{p^1+\ldots+p^{m-1}+k} \gamma_i w_i \quad \text{with } \gamma_i = \sum_{i=1}^{p^1+\ldots+p^{m-1}+k} \frac{(g_{k+1}, Cw_i)}{(Cw_i, w_i)} \tag{4.27}$$

5 Parametric Studies

The restarting techniques can be interpretated as a Krylov preconditioner, when the matrix C changes. This situation occurs when solving nonlinear problem [Rog93]

[Rou90]. We propose to use the Krylov preconditioner in another context. Suppose we have computed the eigenpairs of a structure, and that later a new system has to be solved coming from a small modification of this structure:

$$C^{mod}\mu = b. \tag{5.28}$$

The existing preconditioner P (Dirichlet or lumped preconditioner in the case of the dual Schur complement method) is then improved by the Krylov space which comes from the previous computation (p_1 search directions w and matrix-vector products Cw) with matrix C. It is given as follows:

$$P^{new}g_{k+1} = Pg_{k+1} + \sum_{i=1}^{p_1} \frac{(g_{k+1}, w_i) - (Pg_{k+1}, Cwi)}{(Cw_i, w_i)}\, w_i. \tag{5.29}$$

The technique gives good results if the spectrum of C and C^{mod} are close. Consequently the modification must be done far from the interface between substructures.

6 Numerical Results

We study a steel three-dimensional cantilever beam ($20m$×$4m$×$4m$). The finite element discretization is done with 6,400 hexahedral Q1-Lagrange elements (23,595 dof). The beam is cut in 8 slices (1×1×8), each substructure has 3,267 dof and the interface has 2,541 dof. Ten eigenpairs are required. The computation is carried out on 8 nodes of the Intel Paragon. Figure (1) shows the iteration history with different acceleration techniques. The restarting technique reduces dramatically the iteration number. We note that due to the particular decomposition (no cross points), the improvement is less important in case of the Dirichlet preconditioner. The three times appearing in the legend to the figure (1), correspond to the total CPU time, the time of the preparation step (assembly and factorization of the local stiffness matrix), and the time spent in the Lanczos and the dual Schur complement methods. In practice, the dual Schur complement method represents almost 80% of this last CPU time. The CPU times point out the best result for the lumped preconditioner. The Krylov preconditioner, figure (2) is tested on a structure 3% longer than the previous one (only one substructure has been modified). The reuse of the Krylov preconditioner reduces the CPU time (-20%). This is a useful numerical tool for parametric studies under conditions pointed out in section 5.

The beam is now box partitioned into 8 (2×2×2), 16 (2×2×4) and 32 (2×2×8) subdomains. Figure (3) and (4) show the numerical and parallel scalability of the method proposed. The CPU time is less for boxes decomposition than it was for slices because of the smaller bandwidth of the local problems.

The Schur dual complement method with coarse grid is insensitive to the number of subdomains. The accuracy of eigenpairs is governed by that of the linear system solution, which must be increased when many eigenpairs are sought.

Figure 1 Restarting technique

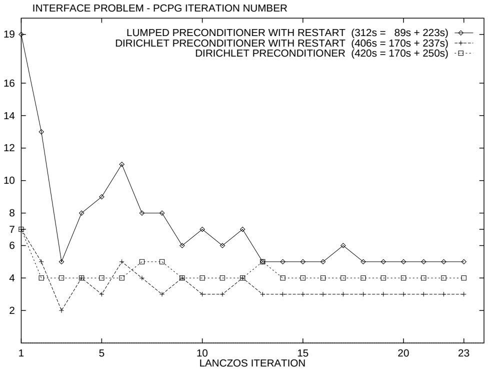


Figure 2 Krylov preconditioner

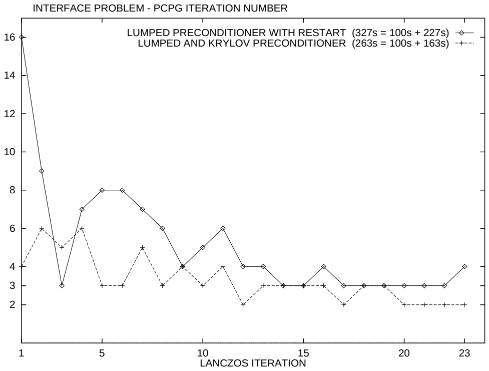


Figure 3 Numerical scalability

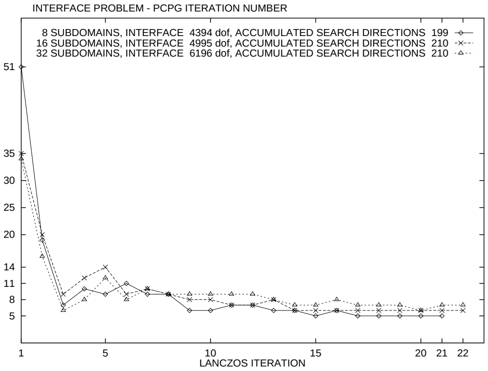
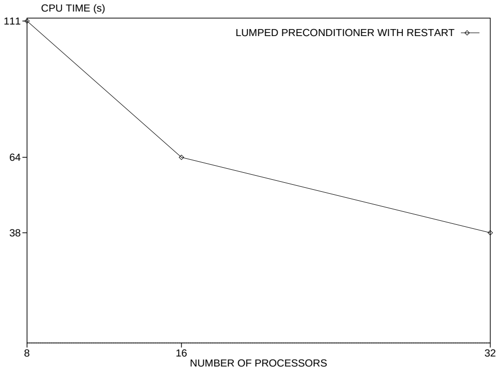


Figure 4 Parallel scalability



7 Conclusion

The method proposed has been tested with success on different examples [Cro97], especially an ill-conditioned problem (steel-elastomer structure) and presents good features for the parallel solution of large scale eigenvalue problems. It can be improved by including new developments in domain decomposition solvers. For the classical shift-and-invert approach $M(K-\sigma M)^{-1}Mq = \frac{1}{(\lambda-\sigma)}Mq$, in which σ is chosen close to the desired eigenvalue λ, a new coarse grid must be introduced, because there are no more floating subdomains. Finally, extension to nonsymmetric eigenproblem provides no difficulties.

REFERENCES

[BKP97] Bramble J., Knyazev A., and Pasciak J. (1997) A subspace preconditioning algorithm for eigenvectors/eigenvalue computation. *Advances in Computational Mathematics* to appear.

[CG82] Carnoy E. and Géradin M. (1982) On the practical use of the Lanczos algorithm in finite element applications to vibration and bifurcation problems. In *Matrix Pencils*, number 973 in Lectures Notes in Mathematics, pages 156–176. Springer Verlag. Proceedings, Pite Havsbad.

[Cro97] Cros J.-M. (to appear 1997) *Résolution de problèmes aux valeurs propres en* *calcul des structures par utilisation du calcul parallèle.* Doctoral dissertation, Ecole Normale Supérieure de Cachan.

[FC95] Farhat C. and Chen P.-S. (1995) Tailoring domain decomposition methods for efficient parallel coarse grid solution and for systems with many right hand sides. In Keyes and Xu [KX95], pages 401–406.

[FR94] Farhat C. and Roux F. (Juin 1994) *Implicit parallel processing in structural mechanics*, volume 2 of *Computational Mechanics Advances.* North-Holland.

[GLS86] Grimes R., Lewis G., and Simon D. (1986) Eigenvalue problems and algorithms in structural engineering. In Cullum J. and Willoughby R. (eds) *Large Scale Eigenvalue Problems.*

[GR94] Géradin M. and Rixen D. (1994) *Mechanical Vibrations.* Wiley.

[KX95] Keyes D. E. and Xu J. (eds) (1995) *Proc. Seventh Int. Conf. on Domain Decomposition Meths.* Number 180 in Contemporary Mathematics. AMS, Providence.

[LT90] Lascaux P. and Theodor R. (1990) *Analyse numérique matricielle appliquée à l'art de l'ingénieur*, volume 1. Masson.

[Rog93] Rogé C. (1993) *Méthodes de calcul de structures composites sur calculateur MIMD.* Doctoral dissertation, Université Paris 6.

[Rou90] Roux F.-X. (1990) Acceleration of the outer conjugate gradient by reorthogonalization for a domain decomposition method for structural analysis problems. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Proc. Third Int. Conf. on Domain Decomposition Meths.*, pages 314–319. SIAM, Philadelphia.

[Rou94] Roux F.-X. (1994) Parallel implementation of domain decomposition method for non-linear elasticity problems. In Keyes D. E., Saad Y., and Trulhar D. G. (eds) *Domain-based Parallelism and Problem Decomposition Methods in Computational Science and Engineering*, pages 161–175. SIAM, Minneapolis.

[RTD95] Roux F.-X. and Tromeur-Dervout D. (1995) Parallelization of a multigrid solver via a domain decomposition method. In Keyes and Xu [KX95], pages 439–444.

[Saa87] Saad Y. (1987) On the Lanczos method for solving symmetric linear systems with several right-hand sides. *Math. Comp.* 48(178): 651–662.

[SVdV96] Sleijpen G. and Van der Vorst H. (1996) A Jacobi-Davidson iteration method for linear eigenvalue. *SIAM J. Matrix Anal. Appl.* 17: 401–425.

38

Domain Decompositions of Wave Problems Using a Mixed Finite Element Method

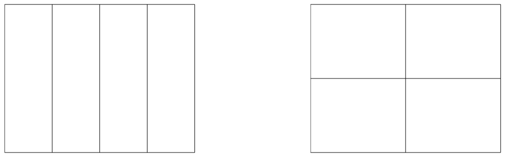
E. J. Dean and R. Glowinski

1 Introduction

In this article we discuss the numerical solution of the *wave equation* by *domain decomposition methods.* Such methods, for the numerical solution of partial differential equations, have become very popular in recent years due to the emergence of parallel computers. While most of the emphasis has been on elliptic and parabolic problems, a few authors ([MS87, Far91, DG93, Dup94]) have considered the hyperbolic case. We will discuss the domain decomposition solution of a non-constant coefficient wave equation, with (first order) absorbing boundary conditions, using a mixed finite element formulation. The mixed formulation, in addition to obtaining accurate gradient approximations, will better handle problems with rapidly varying or discontinuous coefficients. The mixed formulation also allows us to treat both striped and box decompositions (Figure 1) in the same manner. This is in contrast to a conforming method where the intersection of the interfaces, in a box decomposition, can present additional complexity. (This difficulty, and a remedy, is discussed in [DG93].) For the mixed method, interface conditions will be treated by a method combining Lagrange multipliers and a conjugate gradient algorithm. The results of numerical experiments will be presented.

Let Ω be a bounded domain of R^d ($d \geq 1$) with boundary Γ. Motivated by wave

Figure 1. A striped decomposition and a box decomposition.



Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

propagation problems in geophysics, we consider the numerical solution of the following linear wave problem:

$$\rho u_{tt} - \nabla\cdot(a\nabla u) = f \qquad \text{in} \quad \Omega \times (0,T), \tag{1.1}$$

with boundary condition:

$$\sqrt{a\rho}\, u_t + a\nabla u \cdot \mathbf{n} = 0 \qquad \text{on} \quad \Gamma \times (0,T), \tag{1.2}$$

and initial conditions:

$$u(0) = u_0, \;\; u_t(0) = u_1. \tag{1.3}$$

Here $\mathbf{n}$ is the unit outward normal vector on Γ. We will assume that a, ρ are two piecewise continuous functions on $\bar{\Omega}$ satisfying: $a(x) \geq a_0 > 0$, $\rho(x) \geq \rho_0 > 0$.

If we introduce the new variable

$$\mathbf{p} = a\nabla u, \tag{1.4}$$

then it follows from (1.1) and (1.4) that u and $\mathbf{p}$ satisfy the variational equations:

$$\int_\Omega (\rho\, u_{tt} - \nabla\cdot\mathbf{p} - f)\, v\, dx = 0, \qquad \forall v \in L^2(\Omega), \tag{1.5}$$

and

$$\int_\Omega a^{-1}\, \mathbf{p}\cdot\mathbf{q}\, dx + \int_\Omega u\, \nabla\cdot\mathbf{q}\, dx = \int_\Gamma u\, \mathbf{q}\cdot\mathbf{n}\, d\Gamma, \qquad \forall\mathbf{q} \in H(\Omega, div). \tag{1.6}$$

(Here $H(\Omega, div) = \{\mathbf{q} \in (L^2(\Omega))^d : \nabla\cdot\mathbf{q} \in L^2(\Omega)\}$).

We can accommodate the boundary condition (1.2) by differentiating (1.6) in time, and using (1.2), to get

$$\begin{aligned} &\int_\Omega a^{-1}\, \mathbf{p}_t\cdot\mathbf{q}\, dx + \int_\Omega u_t\, \nabla\cdot\mathbf{q}\, dx \\ &\quad + \int_\Gamma (a\rho)^{-\frac{1}{2}}\, (\mathbf{p}\cdot\mathbf{n})\, (\mathbf{q}\cdot\mathbf{n})\, d\Gamma = 0, \qquad \forall\mathbf{q} \in H(\Omega, div). \end{aligned} \tag{1.7}$$

Similarly, we can remove the direct dependence of (1.7) on u_t by differentiating (1.7) in time. By (1.5), and since $\nabla\cdot\mathbf{q} \in L^2(\Omega)$, we get

$$\begin{aligned} &\int_\Omega a^{-1}\mathbf{p}_{tt}\cdot\mathbf{q}\, dx + \int_\Omega \rho^{-1}\, (\nabla\cdot\mathbf{p} + f)\, \nabla\cdot\mathbf{q}\, dx \\ &\quad + \int_\Gamma (a\rho)^{-\frac{1}{2}}\, (\mathbf{p}_t\cdot\mathbf{n})\, (\mathbf{q}\cdot\mathbf{n})\, d\Gamma = 0, \qquad \forall\mathbf{q} \in H(\Omega, div). \end{aligned} \tag{1.8}$$

2 Domain Decomposition

To simplify the discussion, we will partition the domain Ω into only two subdomains Ω_1 and Ω_2, with the interface γ between Ω_1 and Ω_2. We let a_i, ρ_i, and f_i denote the restriction of a, ρ, and f to subdomain Ω_i, $i = 1, 2$, respectively. If $\mathbf{p}_i \in H(\Omega_i, div), i = 1, 2$, then for $\mathbf{p}_i$ to be the restriction of $\mathbf{p} \in H(\Omega, div)$ to Ω_i it is necessary that

$$\mathbf{p}_1 \cdot \mathbf{n}_1 + \mathbf{p}_2 \cdot \mathbf{n}_2 = 0 \tag{2.9}$$

on the interface γ. Here $\mathbf{n}_i$ is the unit outward normal vector on γ for subdomain Ω_i. Using Lagrange multiplier theory, we can enforce the constraint (2.9) by finding a multiplier $\lambda \in \Lambda$ satisfying the following domain decomposition formulation of (1.8):

Find $\{\mathbf{p}_1(t), \mathbf{p}_2(t), \lambda(t)\} \in H(\Omega_1, div) \times H(\Omega_2, div) \times \Lambda$ so that

$$\begin{aligned}\sum_{i=1}^{2}\Big[\int_{\Omega_i} a_i^{-1}\mathbf{p}_{itt} \cdot \mathbf{q}_i \, dx \quad &+ \quad \int_{\Omega_i} \rho_i^{-1}(\nabla\cdot\mathbf{p}_i + f_i)\, \nabla\cdot\mathbf{q}_i \, dx \\ &+ \int_{\Gamma\cap\partial\Omega_i} (a_i\rho_i)^{-1/2}\, (\mathbf{p}_{i_t} \cdot \mathbf{n}_i)\, (\mathbf{q}_i \cdot \mathbf{n}_i)\, d\Gamma \Big] \\ &= \int_{\gamma} \lambda\, (\mathbf{q}_1 \cdot \mathbf{n}_1 + \mathbf{q}_2 \cdot \mathbf{n}_2)\, d\gamma, \\ &\qquad \forall\{\mathbf{q}_1, \mathbf{q}_2\} \in H(\Omega_1, div) \times H(\Omega_2, div),\end{aligned} \tag{2.10}$$

and

$$\int_{\gamma} (\mathbf{p}_1 \cdot \mathbf{n}_1 + \mathbf{p}_2 \cdot \mathbf{n}_2)\, \mu \, d\gamma = 0, \quad \forall \mu \in \Lambda, \text{ a.e. on } (0, T). \tag{2.11}$$

Remark 1: The choice of Λ is a delicate matter (involving spaces such as $H_{00}^{1/2}(\gamma)$). We have implicitly assumed, in (2.9), that $\mathbf{p}_1 \cdot \mathbf{n}_1 + \mathbf{p}_2 \cdot \mathbf{n}_2 \in L^2(\gamma)$, implying that we can take $\Lambda = L^2(\gamma)$. There will not be a problem with this choice in finite dimensions.
Remark 2: The Lagrange multiplier plays the role of u.

Since we only require $u \in L^2(\Omega)$, the restrictions u_i need only to be in $L^2(\Omega_i)$ and satisfy:

Find $\{u_1(t), u_2(t)\} \in L^2(\Omega_1) \times L^2(\Omega_2)$ so that

$$\begin{aligned}\sum_{i=1}^{2}\Big[\int_{\Omega_i} (\rho_i\, u_{itt} - \nabla\cdot\mathbf{p}_i - f_i)\, v_i \, dx\Big] &= 0, \\ \forall\{v_1, v_2\} \in L^2(\Omega_1) \times L^2(\Omega_2),& \text{ a.e. on } (0, T).\end{aligned} \tag{2.12}$$

3 Space and Time Discretization

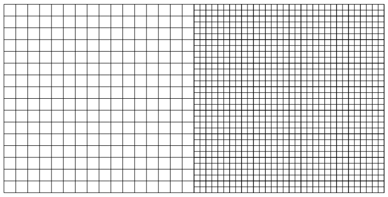
For simplicity we will assume that the spatial dimension $d = 2$ and that the domain Ω, as well as the subdomains $\Omega_i, i = 1, 2$, are rectangles whose boundaries are parallel to the coordinate axes. We will approximate the spaces $H(\Omega_i, div)$ and $L^2(\Omega_i)$ by the lowest order Raviart-Thomas spaces. To this end, we triangulate each rectangle Ω_i into a uniform partition of subrectangles $R_{h_i} = \{K\}$. We will also assume that each rectangle K has edges parallel to the coordinate axes and each rectangle is of uniform size, with h_i the length of the longest side.

We will use the approximation spaces $Q_{h_i} \approx H(\Omega_i, div)$ where

$$Q_{h_i} = \{\mathbf{q}|_K = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} : q_1 = \alpha_K + \beta_K x_1, q_2 = \gamma_K + \delta_K x_2, \forall K \in R_{h_i}\}. \tag{3.13}$$

(Here $x = \binom{x_1}{x_2}$ is a generic point in Ω_i.) We see then that over each rectangle K the vector-valued function $\mathbf{q} \in Q_{h_i}$ will have the first component linear with respect to x_1 and constant with respect to x_2. The situation for the second component of $\mathbf{q}$ is reversed. For the displacement spaces, we will use the approximation $L^2(\Omega_i) \approx V_{h_i} = \{v|_K = \varepsilon_K, \forall K \in R_{h_i}\}$, i.e. the space of piecewise constant functions. We will also assume that the triangulations R_{h_1} and R_{h_2} will be semi-matching at the interface γ as in Figure 2. Finally, the multiplier space Λ is approximated by Λ_h, the space of functions piecewise constant on the edges of the finer triangulation located on γ.

Figure 2. Semi-matching grid.



The time discretization is a domain decomposition implementation of a well known second order *explicit* finite difference scheme for the wave equation. We let $\Delta t (> 0)$ be a time discretization step and let $\mathbf{p}^n_{h_i} \approx \mathbf{p}_i(n\Delta t)$, $u^n_{h_i} \approx u_i(n\Delta t)$, and $\lambda^n_h \approx \lambda(n\Delta t)$, for $i = 1, 2$ and for $n = 0, 1, 2, \ldots$. The full approximate problem to problem (2.10)-(2.12) is:

For $n = 0, 1, 2, \cdots$, find $\{\mathbf{p}^{n+1}_{h_1}, \mathbf{p}^{n+1}_{h_2}, \lambda^n_h\} \in Q_{h_1} \times Q_{h_2} \times \Lambda_h$ so that

$$\begin{aligned} \sum_{i=1}^{2} \Big[\int_{\Omega_i} & a_i^{-1} \frac{\mathbf{p}^{n+1}_{h_i} - 2\mathbf{p}^n_{h_i} + \mathbf{p}^{n-1}_{h_i}}{|\Delta t|^2} \cdot \mathbf{q}_{h_i}\, dx \\ &+ \int_{\Omega_i} \rho_i^{-1} (\nabla \cdot \mathbf{p}^n_{h_i} + f_i)\, \nabla \cdot \mathbf{q}_{h_i}\, dx \\ &+ \int_{\Gamma \cap \partial\Omega_i} (a_i \rho_i)^{-1/2} \left(\frac{\mathbf{p}^{n+1}_{h_i} - \mathbf{p}^{n-1}_{h_i}}{\Delta t} \cdot \mathbf{n}_i \right) (\mathbf{q}_{h_i} \cdot \mathbf{n}_i)\, d\Gamma \Big] \\ &= \int_\gamma \lambda^n_h (\mathbf{q}_{h_1} \cdot \mathbf{n}_1 + \mathbf{q}_{h_2} \cdot \mathbf{n}_2)\, d\gamma, \quad \forall \{\mathbf{q}_{h_1}, \mathbf{q}_{h_2}\} \in Q_{h_1} \times Q_{h_2}, \end{aligned} \tag{3.14}$$

and

$$\int_\gamma (\mathbf{p}_{h_1}^{n+1}\cdot\mathbf{n}_1 + \mathbf{p}_{h_2}^{n+1}\cdot\mathbf{n}_2)\,\mu_h\,d\gamma = 0, \quad \forall\mu_h \in \Lambda_h, \tag{3.15}$$

with $\mathbf{p}_{h_i}^0 = a_i\nabla u_{0h_i}$ and $\mathbf{p}_{h_i}^1 - \mathbf{p}_{h_i}^{-1} = 2\Delta t\; a_i\nabla u_{1h_i}, \quad i = 1,2,$

and find $\{u_{h_1}, u_{h_2}\} \in V_{h_1}\times V_{h_2}$ so that

$$\sum_{i=1}^{2}\Big[\int_{\Omega_i}(\rho_i\,\frac{u_{h_i}^{n+1} - 2u_{h_i}^n + u_{h_i}^{n-1}}{|\Delta t|^2} - \nabla\cdot\mathbf{p}_{h_i}^n - f_i)\,v_{h_i}\,dx\Big] = 0, \tag{3.16}$$
$$\forall\{v_{h_1}, v_{h_2}\}\in V_{h_1}\times V_{h_2},$$

with $u_{h_i}^0 = u_{0h_i}$ and $u_{h_i}^1 - u_{h_i}^{-1} = 2\Delta t\,u_{1h_i}, \quad i = 1,2.$

Notice that (3.14),(3.15) do not depend on the displacement approximation $u_{h_i}^{n+1}$. Hence (3.16) needs to be calculated only if we are interested in approximating the displacements $u(t)$ as well as $\mathbf{p}(t)$. For the applications in which we are interested, the material coefficients a_i and $\rho_i, i = 1,2$, are assumed to be piecewise constant. If we approximate the forcing term f by a piecewise constant interpolant, then all the integrals in (3.14)-(3.16) can be computed exactly using Simpson's rule.

To find $u_{h_i}^{n+1}$ in (3.16) we need only to solve a diagonal linear system. To find $\mathbf{p}_{h_i}^{n+1}$ and λ_h^n in (3.14),(3.15) we solve, at each time step, a system of linear equations of the form

$$\mathbf{A}\hat{\mathbf{p}} + \mathbf{B}^T\hat{\lambda} \;=\; \hat{\mathbf{b}} \tag{3.17}$$
$$\mathbf{B}\hat{\mathbf{p}} \;=\; \hat{\mathbf{c}} \tag{3.18}$$

where $\mathbf{A}\in R^{N\times N}$ is symmetric positive definite and $\mathbf{B}\in R^{M\times N}$ $(M << N)$. Using the Schur Complement we can solve for $\hat{\lambda}$ by solving

$$(\mathbf{B}\mathbf{A}^{-1}\mathbf{B}^T)\hat{\lambda} = \mathbf{B}\mathbf{A}^{-1}\hat{\mathbf{b}} - \hat{\mathbf{c}} \tag{3.19}$$

using, for example, the Conjugate Gradient Algorithm in the form given by Glowinski and LeTallec [GL89]:

```
0) $\hat{\lambda}_0$ is given. ($\hat{\lambda}_0 \equiv \hat{\lambda}^{n-1}$)
   Solve $\mathbf{A}\hat{\mathbf{p}}_0 = \hat{\mathbf{b}} - \mathbf{B}^T\hat{\lambda}_0$.
   Compute $\hat{\mathbf{g}}_0 = \hat{\mathbf{c}} - \mathbf{B}\hat{\mathbf{p}}_0$.
   Set $\hat{\mathbf{w}}_0 = \hat{\mathbf{g}}_0$.

1) For $k = 0,1,2,\dots$ until convergence:
   1.1) Solve $\mathbf{A}\hat{\mathbf{z}}_k = \mathbf{B}^T\hat{\mathbf{w}}_k$.
   1.2) $\rho_k = \frac{|\hat{\mathbf{g}}_k|^2}{(\mathbf{B}\hat{\mathbf{z}}_k,\hat{\mathbf{w}}_k)}$.
```

1.3) $\hat{\lambda}_{k+1} = \hat{\lambda}_k - \rho_k \hat{\mathbf{w}}_k$.

1.4) $\hat{\mathbf{p}}_{k+1} = \hat{\mathbf{p}}_k + \rho_k \hat{\mathbf{z}}_k$.

1.5) $\hat{\mathbf{g}}_{k+1} = \hat{\mathbf{g}}_k - \rho_k \mathbf{B} \hat{\mathbf{z}}_k$.

1.6) $\gamma_k = \frac{|\hat{\mathbf{g}}_{k+1}|^2}{|\hat{\mathbf{g}}_k|^2}$.

1.7) $\hat{\mathbf{w}}_{k+1} = \hat{\mathbf{g}}_{k+1} + \gamma_k \hat{\mathbf{w}}_k$.

We note that the matrix $\mathbf{A}$ is block diagonal with symmetric positive definite tridiagonal blocks so the linear system in steps 0) and 1.1) can be solved very efficiently. We also mention that the entire iterative algorithm is very efficient, usually requiring only 1 or 2 iterations to get a substantial reduction in the relative size of the gradient $\hat{\mathbf{g}}_k$.

4 Numerical Experiments

The experiments discussed here are motivated by applications in geophysics and are related to the numerical simulation of an explosion. To this end, we have taken the forcing term f to be the Ricker pulse (see [BT91]):

$$f(\mathbf{x}, t) = \begin{cases} d(t)s(r), & \text{if } 0 \leq r \leq R \text{ and } 0 \leq t \leq \frac{2}{f_0}, \\ 0, & \text{otherwise,} \end{cases} \tag{4.20}$$

where $s(r) = \left[\frac{3}{\pi} \left(\frac{r^2 - R^2}{R^3} \right)^2 \right]$, $r^2 = (x_1 - x_1^0)^2 + (x_2 - x_2^0)^2$, $d(t) = A(1 - 2\tau^2)e^{-\tau^2}$, and $\tau = \pi(f_0 t - 1)$. In (4.20) $s(r)$ is meant to approximate the Dirac measure centered at the point (x_1^0, x_2^0). Here R, A, and f_0 are the radius, amplitude, and frequency parameters for the pulse. We also used the initial conditions $u_0 = u_1 = 0$.

In Figure 3 we see the evolution of the wave over four subdomains arranged in a 2×2 partition of Ω ($\Omega_1 = (0, 0.5) \times (0, 0.5), \Omega_2 = (0, 0.5) \times (0.5, 1), \Omega_3 = (0.5, 1) \times (0, 0.5), \Omega_4 = (0.5, 1) \times (0.5, 1)$.) The discretization was identical in all four subdomains ($h_1 = h_2 = h_3 = h_4$) with matching grids at the interfaces. We notice that there is no deformation as the wave front passes through the interfaces. We should also note that no special arrangements have to be made at the crossing point $(0.5, 0.5)$ since the Lagrange multipliers are discontinuous there. This is in contrast to the conforming method presented in [DG93].

In Figure 4 we compare a global calculation of the wave propagation with the domain decomposition method. In the domain decomposition calculation we have partitioned Ω into two subdomains ($\Omega_1 = (0, 1) \times (0, 1), \Omega_2 = (1, 2) \times (0, 1)$) where the discretization parameters satisfy $h_2 = h_1/2$ and the grids are semi-matching at the interface as in Figure 2. The material constants are equal in both subdomains. The pulse was centered at the point $(0.5, 0.5)$ and the figure shows the remnants of the wave fronts at a time when the front has passed the interface γ. We notice that the wave fronts are almost identical.

In Figure 5 we have the same domain decomposition described for Figure 4. The material constants in this case satisfy $a_1 = 4a_2$ and $\rho_1 = \rho_2$, so the wave is propagating twice as fast in Ω_1 as in Ω_2. After 100 time steps, we see the wave front about to intersect the interface γ. After 200 time steps, we see the original wave front still developing in Ω_2, with a reflection propagating in the opposite direction.

Figure 3. Four subdomains.

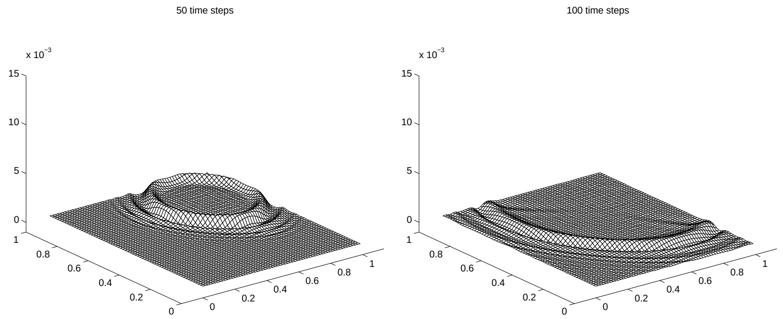


Figure 4. Global versus domain decomposition solutions.

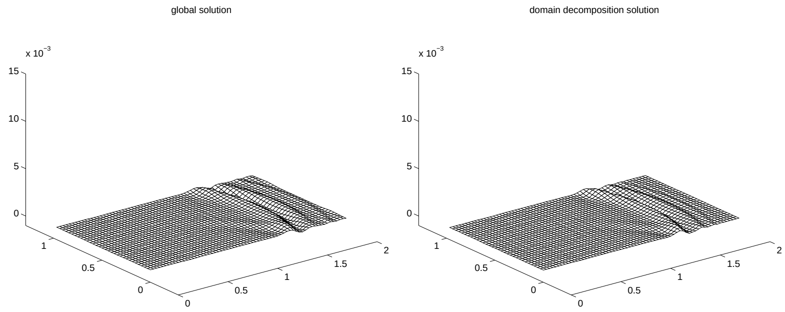
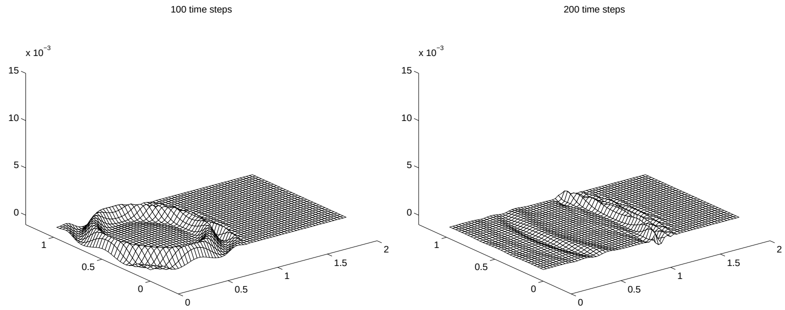


Figure 5. Subdomains with different material constants.



Acknowledgement

We would like to acknowledge the helpful comments and suggestions of the following individuals: T.-W. Pan, T. A. Parks, R. Sanders, M. F. Wheeler, and I. Yotov. The support of the Texas Board of Higher Education (Grants 003652-146ATP and 003652-091ARP) and the National Science Foundation (Grant DMA-9696177) is also acknowledged.

REFERENCES

[BT91] Bamberger A. and Tran Q. H. (1991) Propagation and attenuation of a Ricker pulse in one-dimensional randomly heterogeneous media: A numerical study. In Glowinski R. (ed) *Computing Methods in Applied Sciences and Engineering*. Nova Science, Commack, New York.

[DG93] Dean E. J. and Glowinski R. (1993) A domain decomposition method for the wave equation. In Horowitz J. and Lions J. L. (eds) *Les Grands Systemes des Sciences et de la Technologie*. Masson, Paris.

[Dup94] Dupont T. (1994) Non-iterative domain decomposition for second order hyperbolic problems. In Quarteroni A., Périaux J., Kuznetsov Y. A., and Widlund O. B. (eds) *Proc. Sixth Int. Conf. on Domain Decomposition Meths.* AMS, Providence.

[Far91] Farhat C. (1991) Parallel processing in structural mechanics: Blending mathematical, implementational and technological advances. In *Computing Methods in Applied Sciences and Engineering*. Nova Science, Commack, New York.

[GL89] Glowinski R. and LeTallec P. (1989) *Augmented Lagrangian And Operator Splitting Methods in Nonlinear Mechanics*. SIAM, Philadelphia.

[MS87] Meza J. C. and Symes W. W. (1987) Domain decomposition algorithms for linear hyperbolic equations. Technical Report 87-20, Rice University, Department of Mathematical Sciences, Rice University, Houston, Texas.

39

ADN and ARN Domain Decomposition Methods for Advection-Diffusion Equations

F. Gastaldi, L. Gastaldi, and A. Quarteroni

1 Introduction

We consider adaptive domain decomposition methods for the solution of advection-diffusion boundary value problems.

The computational domain is partitioned into disjoint subdomains that do not overlap. The original boundary value problem is reformulated in a split form on the subdomains, and the subdomain solutions satisfy suitable matching conditions at subdomain interfaces. These transmission conditions are then used to set up iterative procedures among subdomains. In this work we review a family of methods, known as ADN (Adaptive Dirichlet Neumann) and ARN (Adaptive Robin Neumann), which were previously introduced in [CQ95], [Cic], and [Tro96].

The idea behind these methods is to choose interface conditions which are compatible with those of the hyperbolic problem obtained letting the diffusion coefficient go to zero. Then iterative methods are introduced splitting the above interface conditions in a way which is adapted to the local flow direction. This prevents the rise of artificial layers at subdomain interfaces as the advection becomes dominant. An extensive analysis of the properties enjoyed by these methods is carried out in [GGQ96]. In this work we report briefly the main results of this analysis.

2 Advection-diffusion Boundary Value Problem

Let Ω be a bounded, connected, open subset of $\mathbb{R}^2$ with a Lipschitz continuous boundary $\partial\Omega$. Let $\epsilon > 0$ be a constant diffusion coefficient, $\mathbf{b} = \mathbf{b}(x)$ denote the given flow velocity and $b_0 = b_0(x)$ be an absorption coefficient. The boundary value

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

problem we are considering reads: Find u such that

$$\left\{ \begin{array}{rcll} L_\epsilon u := -\epsilon\Delta u + \operatorname{div}(\mathbf{b}u) + b_0 u & = & f & \text{in } \Omega, \\ u & = & 0 & \text{on } \partial\Omega, \end{array} \right. \tag{2.1}$$

where $f = f(x)$ is a given body force.

The characteristic quantity $\omega = (2\epsilon)^{-1}\|\mathbf{b}\|_\infty$ (essentially the analog of the Reynolds number for Navier–Stokes equations) will be used; in particular, we will be primarily concerned with the case $\omega \gg 1$ (advection-dominated).

The method can be applied in the case of several subdomains (see Remark 1 below); for simplicity, here we restrict the discussion to the model case where the domain is divided into two non–overlapping subdomains. Let Ω_1 and Ω_2 be these subdomains, whose boundaries $\partial\Omega_1$ and $\partial\Omega_2$ are supposed to be Lipschitz continuous. The common interface $\partial\Omega_1 \cap \partial\Omega_2$ is denoted by Γ; the normal unit vector on Γ pointing into Ω_2 is denoted by $\mathbf{n}$. We assume that Γ is piecewise $\mathbf{C}^1$ and we distinguish three subsets of the regular part of Γ (namely, where $\mathbf{n}$ exists):

$$\left\{ \begin{array}{rcl} \Gamma^0 & := & \{x \in \Gamma : \mathbf{b}(x) \cdot \mathbf{n}(x) = 0\}, \\ \Gamma^{in} & := & \{x \in \Gamma : \mathbf{b}(x) \cdot \mathbf{n}(x) < 0\}, \\ \Gamma^{out} & := & \{x \in \Gamma : \mathbf{b}(x) \cdot \mathbf{n}(x) > 0\}, \end{array} \right. \tag{2.2}$$

which are identified through the local direction of the flow field $\mathbf{b}(x)$ at the subdomain interface.

The original boundary value problem (2.1) can be reformulated as follows. Denoting by u_1 and u_2 the restriction of the solution u to the subdomains Ω_1 and Ω_2, respectively, it can be shown that u_1 and u_2 satisfy the split problem:

$$L_\epsilon u_1 = f \quad \text{in} \quad \Omega_1, \tag{2.3}$$

$$L_\epsilon u_2 = f \quad \text{in} \quad \Omega_2, \tag{2.4}$$

$$u_i = 0 \quad \text{on} \quad \partial\Omega_i \backslash \Gamma \quad i = 1, 2. \tag{2.5}$$

$$u_1 = u_2 \quad \text{on} \quad \Gamma, \quad (D) \tag{2.6}$$

$$\epsilon\frac{\partial u_1}{\partial \mathbf{n}} = \epsilon\frac{\partial u_2}{\partial \mathbf{n}} \quad \text{on} \quad \Gamma. \quad (N) \tag{2.7}$$

The interface matching conditions (2.6) and (2.7) enforce the simultaneous continuity of the subdomain solutions (Dirichlet condition, say D) and of their normal derivatives (Neumann condition, say N). Besides the *Dirichlet–Neumann* formulation of the transmission conditions one could use as well the continuity of the fluxes (Robin condition, say R) and combine it either with D or with N. For example (2.6) and (2.7) can be replaced equivalently by the *Robin–Neumann* matching conditions:

$$\epsilon\frac{\partial u_1}{\partial \mathbf{n}} - \mathbf{b}\cdot\mathbf{n}u_1 = \epsilon\frac{\partial u_2}{\partial \mathbf{n}} - \mathbf{b}\cdot\mathbf{n}u_2 \quad \text{on} \quad \Gamma, \quad (R) \tag{2.8}$$

$$\epsilon\frac{\partial u_1}{\partial \mathbf{n}} = \epsilon\frac{\partial u_2}{\partial \mathbf{n}} \quad \text{on} \quad \Gamma, \quad (N) \tag{2.9}$$

provided $meas\{x \in \Gamma : \mathbf{b}(x) \cdot \mathbf{n}(x) = 0\} = 0$.

For brevity, we restrict our discussion to these two types of transmission conditions. For the analysis of other conditions we refer to [GGQ96].

3 Iterative Algorithms for Domain Decomposition

The next step is to set up iterative procedures between the subdomains, based on the problem splitting (2.3)–(2.5) with D-N ((2.6)–(2.7)) or R-N ((2.8)–(2.9)) interface conditions. We define a sequence $\{u_1^n, u_2^n\}$, where u_1^n satisfies (2.3) and (2.5), u_2^n satisfies (2.4) and (2.5) along with either type of boundary conditions at the subdomain interface Γ. A first option is given by the standard Dirichlet–Neumann iterative algorithm: Given u_i^0 in Ω, $(i=1,2)$ solve for each $n \geq 1$

$$(D)\left\{\begin{array}{rcll} L_\epsilon u_1^n & = & f & \text{in } \Omega_1 \\ u_1^n & = & 0 & \text{on } \partial\Omega_1\backslash\Gamma \\ u_1^n & = & \lambda^{n-1} & \text{on } \Gamma \end{array}\right. \qquad (N)\left\{\begin{array}{rcll} L_\epsilon u_2^n & = & f & \text{in } \Omega_2 \\ u_2^n & = & 0 & \text{on } \partial\Omega_2\backslash\Gamma \\ \epsilon\frac{\partial u_2^n}{\partial \mathbf{n}} & = & \epsilon\frac{\partial u_1^n}{\partial \mathbf{n}} & \text{on } \Gamma \end{array}\right. ,$$

where $\lambda^{n-1} = \theta u_2^{n-1} + (1-\theta)u_1^{n-1}$ and θ is a relaxation parameter.

Of course, the conditions at the interface can be interchanged; hence one can also solve first the problem in Ω_1 with Neumann condition at the interface and then the problem in Ω_2 with Dirichlet conditions at the interface. The two choices are suitable for viscous-dominated flows where the skew symmetric part of the operator is not too big. Well known results on such schemes can be found in the papers by Bjørstad and Widlund [BW86] and by Marini and Quarteroni [MQ89], where relaxation is proven to be essential for convergence. But for advection-dominated problems the conditions at the interface have to be set up carefully.

Indeed, the interface conditions have to be adapted to the orientation of the transport field across the interface. The reason why Neumann conditions are used on outflow boundaries is that a Dirichlet condition prescribing specific values for the solution could generate artificial internal layers whose steepness is proportional to ω. Similar comments hold also for Robin–Neumann matching conditions with Robin playing the same role as Dirichlet. Therefore we consider the following algorithm: given u_i^0 in Ω_i, $(i=1,2)$, solve for each $n \geq 1$

$$\left\{\begin{array}{rcll} L_\epsilon u_1^n & = & f & \text{in } \Omega_1 \\ u_1^n & = & 0 & \text{on } \partial\Omega_1\backslash\Gamma \\ \psi(u_1^n) & = & \lambda^{n-1} & \text{on } \Gamma^{in} \\ \epsilon\frac{\partial u_1^n}{\partial \mathbf{n}} & = & \epsilon\frac{\partial u_2^{n-1}}{\partial \mathbf{n}} & \text{on } \Gamma^{out} \end{array}\right. \quad \left\{\begin{array}{rcll} L_\epsilon u_2^n & = & f & \text{in } \Omega_2 \\ u_2^n & = & 0 & \text{on } \partial\Omega_2\backslash\Gamma \\ \psi(u_2^n) & = & \mu^n & \text{on } \Gamma^{out} \\ \epsilon\frac{\partial u_2^n}{\partial \mathbf{n}} & = & \epsilon\frac{\partial u_1^n}{\partial \mathbf{n}} & \text{on } \Gamma^{in} \end{array}\right. , \tag{3.10}$$

with

$$\psi(v) := \left\{\begin{array}{ll} v & \text{for ADN method} \\ \epsilon\frac{\partial v}{\partial \mathbf{n}} - \mathbf{b}\cdot\mathbf{n}v & \text{for ARN method} \end{array}\right. , \tag{3.11}$$

$$\lambda^{n-1} := \theta'\psi(u_2^{n-1}) + (1-\theta')\psi(u_1^{n-1}), \tag{3.12}$$

$$\mu^n := \theta''\psi(u_1^n) + (1-\theta'')\psi(u_2^{n-1}), \tag{3.13}$$

θ' and θ'' being two real parameters that are used to accelerate the convergence of the iterative procedure. Notice that the condition on the interface is now split into two parts: Along the outflow part of the interface we enforce the continuity of the normal derivative, while along the inflow part we enforce the continuity of the trace for Adaptive Dirichlet–Neumann algorithm (ADN) or the continuity of the flux for Adaptive Robin–Neumann algorithm (ARN).

The parameters θ' and θ'' in (3.12) and (3.13) are used to guarantee possible *under-relaxation* when needed to ensure convergence. Typically, a single parameter θ $(= \theta' = \theta'')$ suffices (two parameters allow better flexibility in pursuing an optimal criterion), and often $\theta = 1$ (no relaxation) is a very good choice.

Finally, let us briefly introduce the damped version of the Adaptive Robin–Neumann method (denoted by d–ARN). It consists in substituting the condition that enforces the continuity of the normal derivative on Γ^{out} with a homogeneous Neumann condition. Hence the d–ARN algorithm consists in solving the sub-problem in Ω_1 provided the condition along the outflow part of Γ is replaced by the following equation: $\epsilon\partial u_1^n/\partial\mathbf{n} = 0$ on Γ^{out}. Similarly, one solves the sub– problem in Ω_2 with the damped condition: $\epsilon\partial u_2^n/\partial\mathbf{n} = 0$ on Γ^{in}. The main reason for introducing the damped form of our algorithms is that the damped forms weaken the coupling between u_1^n and u_2^n at the interface Γ, so that the convergence of the corresponding algorithm is faster in general. In particular, when the flow field has a constant direction at the subdomain interface, i.e., $\mathbf{b}(x)\cdot\mathbf{n}(x)$ is either always positive or always negative on Γ, then a single iteration is enough to solve the given problem. This introduces an error, which can be proved not to grow at each iteration, so that the method produces a sequence which is weakly convergent. Moreover, the solution we get for n going to infinity is not too far from the exact one and the error can be measured in terms of a suitable norm of the normal derivative of the exact solution along the interface multiplied by $\sqrt{\epsilon}$. Therefore if the interface is far from any layer (hence the normal derivative is bounded independently of ϵ), then we obtain an error bound of order $\sqrt{\epsilon}$.

The convergence of these methods can be proven working out the error equations, for the complete analysis see [GGQ96]. By subtracting the iterative solution at step n from the exact solution we obtain the errors $e_1^n = u_1 - u_1^n$ and $e_2^n = u_2 - u_2^n$, which solve the same problems as before with homogeneous data inside Ω_1 and Ω_2, respectively.

For the ARN method without relaxation we have obtained the following estimate

$$\|e_2^n\|_\Gamma \le \|e_2^{n-1}\|_\Gamma \quad \forall n, \tag{3.14}$$

where

$$\|e_2^n\|_\Gamma^2 = \int_{\Gamma^{in}} \frac{1}{|\mathbf{b}\cdot\mathbf{n}|}\left(\epsilon\frac{\partial e_2^n}{\partial\mathbf{n}} - \mathbf{b}\cdot\mathbf{n}e_2^n\right)^2 ds \quad + \int_{\Gamma^{out}} \frac{1}{\mathbf{b}\cdot\mathbf{n}}\left(\epsilon\frac{\partial e_2^n}{\partial\mathbf{n}}\right)^2 ds. \tag{3.15}$$

The formula (3.14) expresses that the error at the interface, measured in the norm (3.15), does not grow at each iteration. Moreover, the estimate (3.14) implies a weak convergence of the sequence $\{u_1^n, u_2^n\}$ as it is stated in the following lemma:

Lemma 1 *The sequence* $\{u_1^n, u_2^n\}$ *converges weakly in* $\mathbf{H}^1(\Omega_1)\times\mathbf{H}^1(\Omega_2)$ *to* $\{u_1, u_2\}$ *solution of (2.3)–(2.5), (2.8)–(2.9).*

However, this result cannot provide any useful information for example on the speed of the convergence. Therefore we will detail our analysis in a sample problem (see next section).

Remark 1 *An extensive experimental analysis of these methods in the framework of several kinds of numerical realizations (finite elements, finite volumes, spectral methods) and of decompositions using more than two subdomains, including crosspoints, is carried out in [CQ95], [Cic], and [Tro96]. Obviously, when dealing with more than two subdomains, the Robin or Neumann conditions have to be imposed along each interface according to the local direction of the transport field. The analog of the estimate (3.14) can be still obtained, no matter whether there are crosspoints or not. The iterative algorithm can be performed similarly in the ADN framework.*

4 Analysis of a Two-dimensional Case with Constant Transport

Let Ω be the unit square $]0,1[\times]0,1[$, divided into $\Omega_1 =]0,\gamma[\times]0,1[$, $\Omega_2 =]\gamma,1[\times]0,1[$, where $\gamma \in]0,1[$ is given. The interface is the set $\Gamma = \{\gamma\}\times]0,1[$. The transport field is $\mathbf{b} = (b,0)$, with b a positive constant and we take b_0 a non-negative constant. Hence the problem reads:

$$\left\{\begin{array}{rcll} L_\epsilon u := -\epsilon\Delta u + bu_x + b_0 u &=& f & \text{in } \Omega \\ u &=& 0 & \text{on } \partial\Omega. \end{array}\right. \tag{4.16}$$

We construct the sequence $\{u_1^n, u_2^n\}$ as in (3.10)–(3.13) keeping in mind that Γ^{out} coincides with Γ. Then introducing as before the errors $e_i^n = u_i - u_i^n$, $i = 1,2$, we construct a sequence in the following way: Given a function g_1^{n-1}, solve

$$(N)\left\{\begin{array}{rcll} L_\epsilon e_1^n &=& 0 & \text{in } \Omega_1 \\ e_1^n &=& 0 & \text{on } \partial\Omega_1\backslash\Gamma \\ \epsilon e_{1x}^n &=& \epsilon g_1^{n-1} & \text{on } \Gamma, \end{array}\right.$$

then set $g_2^n = \theta\psi(e_1^n) + (1-\theta)\psi(e_2^{n-1})$ on Γ and solve

$$(R) \text{ or } (D)\left\{\begin{array}{rcll} L_\epsilon e_2^n &=& 0 & \text{in } \Omega_2 \\ e_2^n &=& 0 & \text{on } \partial\Omega_2\backslash\Gamma \\ \psi(e_{2x}^n) &=& g_2^n & \text{on } \Gamma \end{array}\right. ,$$

and finally, set $g_1^n = e_{2x}^n$ on Γ.

When $\theta = 1$ (no relaxation), then by separation of variables we obtain

$$g_1^n(y) = \sum_{k=1}^{\infty} \eta_k^n Y_k(y) \quad \forall n \geq 0, \tag{4.17}$$

where Y_k are the eigenfunctions of the induced spectral problem with respect to y: $-\epsilon Y_k'' = \lambda_k Y_k$ in $(0,1)$, $Y_k(0) = Y_k(1) = 0$.

The coefficients of the expansion (4.17) satisfy a recursive formula

$$\eta_k^n = \rho_k \eta_k^{n-1}, \tag{4.18}$$

with ρ_k given by

$$\rho_k^{ARN} = \frac{\tau_k \coth(\tau_k\gamma) - \omega}{\tau_k \coth(\tau_k\gamma) + \omega} \, \frac{\tau_k \coth(\tau_k(1-\gamma)) - \omega}{\tau_k \coth(\tau_k(1-\gamma)) + \omega},$$

$$\rho_k^{ADN} = -\frac{\tau_k \coth(\tau_k(1-\gamma)) - \omega}{\tau_k \coth(\tau_k \gamma) + \omega},$$

and $\omega = b/(2\epsilon)$, $\tau_k = \sqrt{b^2 + 4\epsilon(b_0 + \lambda_k)}/(2\epsilon)$.

If $\theta \neq 1$ then the recursive formula (4.18) holds with ρ_k replaced by $\rho_k^\theta = 1-\theta(1-\rho_k)$. To have the convergence of the method one should show that the sequence g_1^n converges to 0, but the convergence of g_1^n corresponds to the convergence of the sequences $\eta^n := \{\eta_k^n\}_{k\geq 1}$ in the Hilbert space ℓ^2. Owing to (4.18) it turns out that a sufficient condition is that

$$\sup_k |\rho_k| < 1 \tag{4.19}$$

It is not difficult to prove that $0 < \rho_k^{ARN} < 1$ for all k. Moreover, ρ_k^{ARN} is an increasing function of k and converges to 1 as $k \to +\infty$. On the other hand ρ_k^{ADN} is negative. Moreover for γ not too close to 1 we obtain that $-1 < \rho_k^{ADN} < 0$ for all k and the function ρ_k^{ADN} is decreasing and converges to -1 as $k \to +\infty$.

In Fig. 1 the graphs of ρ_k^{ARN} as a function of k for different values of ϵ are drawn (the solid line corresponds to $\epsilon = 10^{-2}$, the dotted line to $\epsilon = 10^{-3}$, the dashed-dotted line to $\epsilon = 10^{-4}$, the dashed line to $\epsilon = 10^{-5}$). Fig. 2 shows the graphs of ρ_k^{ADN} when γ is not too big, for the same values of ϵ.

Fig.1 ρ_k^{ARN} for several values of ϵ

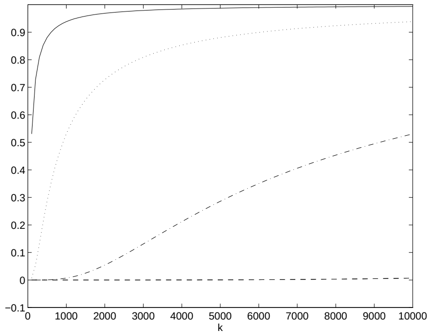
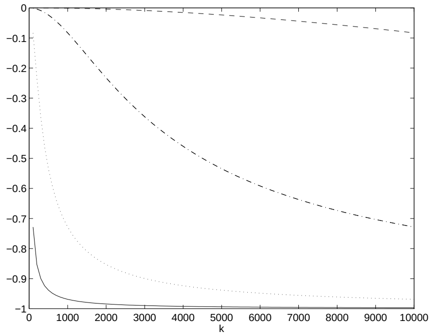


Fig.2 ρ_k^{ADN} for several values of ϵ



Notice that both methods ARN and ADN do not fulfill the sufficient condition (4.19) for convergence. A cure could be the use of a relaxation strategy. It is easy to see that for ARN method ρ_k^θ converge to 1 as $k \to +\infty$, while for ADN it is possible to choose the parameter θ in order to make the sequence converge. However, the formulas and the pictures show that if the high frequency modes are neglected then the two methods provide good convergence for $\theta = 1$. This is in agreement with the numerical results presented in [Tro97]. As a matter of fact a finite dimensional approximation uses only a finite number of modes, as will be shown in the next section.

5 Approximation by Finite Elements

The methods discussed in Sections 2 and 3 are naturally rephrased in a weak form (see [GGQ96]), which is most convenient for finite element approximations. When there are several subdomains, crosspoints may show up. In this case, the matching conditions involving derivatives are naturally enforced through the test functions associated with the crosspoints (see [CQ95, Tro96], where extensive numerical validation is carried out).

The convergence analysis developed in Section 4 can be adapted to the finite element approximation using bilinear finite elements. Let us define the discrete iteration errors e^n_{1h} and e^n_{2h} as the difference between the discrete one–domain solution and the n-th discrete iterate. These errors can be written, by separation of variables, as sums of tensor products

$$e^n_{1h} = \sum_{k=1}^{N_y} \mu^{(n)}_{kh} X_{kh}(x) Y_{kh}(y), \quad e^n_{2h} = \sum_{k=1}^{N_y} \eta^{(n)}_{kh} Z_{kh}(x) Y_{kh}(y),$$

where Y_{kh} are the piecewise linear eigensolutions in the y variable, X_{kh} and Z_{kh} are the piecewise linear solutions of the associated problems in the x variable on $(0, \gamma)$ and $(\gamma, 1)$, respectively. Then the coefficients of these linear combinations satisfy a recursive relation $\eta^{(n)}_{kh} = \rho_{kh} \eta^{(n-1)}_{kh}$, with the reduction factor satisfying $0 < \rho_{kh} < 1$. Hence the sufficient condition (4.19) is satisfied and the discrete iterative procedure converges. In this case the introduction of a relaxation strategy can improve the speed of convergence with a suitable choice of the relaxation parameter.

6 Conclusions

For advection-dominated problems, the adapted iterative algorithms presented have good convergence properties, when a finite number of modes are taken into account along the interface. We observe that this happens when finite dimensional discretizations of the problem are considered. The damped version of the methods is very efficient, with a reasonable choice for the location of the interface.

Due to space limitations, we do not address the issue of efficient implementation (the interested reader can refer to [GGQ96], sect. 1.4): in particular, when using a very large number of subdomains, a coarse grid solver (based on the same adaptive principle) is required in order that the algorithm is scalable.

Acknowledgement

The third author acknowledges the financial support of the Sardinian Regional Authorities.

REFERENCES

[BW86] Bjørstad P. and Widlund O. B. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal.* 23: 1097–1120.

[Cic] Ciccoli M. C.Adaptive domain decomposition algorithms and finite volume/finite element approximation for advection-diffusion equations. Submitted to Journal of Scientific Computing.

[CQ95] Carlenzoli C. and Quarteroni A. (1995) Adaptive domain decomposition methods for advection - diffusion problems. In Babuska I. e. a. (ed) *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, volume 75 of *IMA Volumes in Mathematics and its Applications*, pages 165–199. Springer verlag edition.

[GGQ96] Gastaldi F., Gastaldi L., and Quarteroni A. (1996) Adaptive domain decomposition methods for advection dominated equations. *East-West J. Numer. Math.* 4: 165–206.

[MQ89] Marini L. D. and Quarteroni A. (1989) A relaxation procedure for domain decomposition methods using finite elements. *Numer. Math.* 55: 575–598.

[Tro96] Trotta R. (1996) Multidomain finite elements for advection-diffusion equations. *Appl. Numer. Math.* 21: 91–118.

[Tro97] Trotta L. (1997) Multidomain finite volumes and finite elements for advection–diffusion equations. In Bjørstad P., Espedal M., and Keyes D. (eds) *DD9 Proceedings*.

40

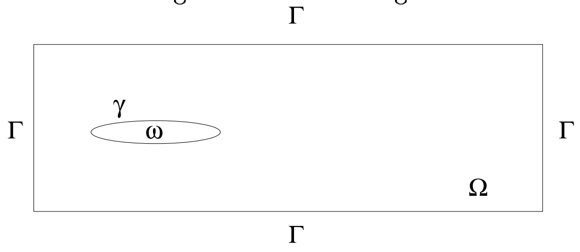
On a Domain Embedding Method for Flow around Moving Rigid Bodies

R. Glowinski, T.-W. Pan, and J. Périaux

1 Introduction

Several applications lead to the numerical simulation of incompressible viscous flow around moving rigid bodies; let us mention for example blood flow around artificial heart valves. In this article we consider only the case where the rigid body motions are known a priori; the more complicated case where the rigid body motions are caused by hydrodynamical forces, among other forces, will be discussed in a forthcoming article. Following an approach advocated — to our knowledge — by Peskin [Pes72] we use a *domain embedding* method (also called *fictitious domain method* by some authors) which consists of filling the moving bodies by the surrounding fluid and taking into account the boundary conditions on these bodies by introducing a well chosen distribution of boundary forces. In the particular case of the *Dirichlet boundary* conditions considered in this article it is quite convenient to use a *Lagrange multiplier* method which is well suited to the variational formulations commonly used to study the Navier-Stokes equations and their approximation, by finite element methods for example. Another important component of the solution method is a time discretization by operator splitting which reduces the simulation to a sequence of subproblems for which efficient solution methods exist already.

Figure 1. The flow region



Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 A Model Problem and its Lagrange Multiplier/Domain Embedding Formulation

The geometrical situation is as in Figure 1. With $\omega = \omega(t)$ a moving rigid body ($\omega \subset \Omega \subset \boldsymbol{R}^d$, d =2, 3), we consider for $t > 0$ the solution of the *Navier-Stokes equations*

$$\frac{\partial \mathbf{u}}{\partial t} - \nu\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p = \mathbf{f}\, in\, \Omega \setminus \overline{\omega(t)}, \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0\, in\, \Omega \setminus \overline{\omega(t)}, \tag{2}$$

$$\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x}), \quad \mathbf{x} \in \Omega \setminus \overline{\omega(0)}, (with\, \nabla\cdot\mathbf{u}_0 = 0), \tag{3}$$

$$\mathbf{u} = \mathbf{g}_0\, on\, \Gamma, \tag{4}$$

$$\mathbf{u} = \mathbf{g}_1\, on\, \gamma(t). \tag{5}$$

In (1)-(5) $\mathbf{u}$ and p denote, as usual, the *velocity* and *pressure*, respectively; $\nu(> 0)$ is the *viscosity*, $\mathbf{f}$ the density of external forces, $\mathbf{x}$ the generic point of $\boldsymbol{R}^d$ ($\mathbf{x} = \{x_i\}_{i=1}^d$), $\gamma(t) = \partial\omega(t)$ and $(\mathbf{u}\cdot\nabla)\mathbf{u} = \{\sum_{j=1}^d u_j \frac{\partial u_i}{\partial x_j}\}_{i=1}^d$. We suppose that $\mathbf{g}_1$ is the velocity on $\gamma(t)$ of the rigid body $\omega(t)$ which implies that $\displaystyle\int_{\gamma(t)} \mathbf{g}_1 \cdot \mathbf{n}\, d\gamma = 0$, and that $\displaystyle\int_\Gamma \mathbf{g}_0 \cdot \mathbf{n}\, d\Gamma = 0$. In the following, we shall use, if necessary, the notation $\phi(t)$ for the function $\mathbf{x} \to \phi(\mathbf{x}, t)$.

We introduce first the functional spaces $\mathbf{V}_{\mathbf{g}_0(t)} = \{\mathbf{v}|\mathbf{v} \in (H^1(\Omega))^d, \mathbf{v} = \mathbf{g}_0(t)\, on\, \Gamma\}$, $\mathbf{V}_0 = (H_0^1(\Omega))^d, L_0^2(\Omega) = \{q|q \in L^2(\Omega); \displaystyle\int_\Omega q\, d\mathbf{x} = 0\}$ and $\Lambda(t) = (\mathbf{H}^{-1/2}(\gamma(t)))^d$. With $\tilde{\mathbf{f}}$ an L^2-lifting of $\mathbf{f}$ in Ω (we can take $\tilde{\mathbf{f}}|_{\overline{\omega(t)}} = \mathbf{0}$) and $\nabla\cdot\mathbf{U}_0 = 0$ ($\mathbf{U}_0|_{\Omega\setminus\overline{\omega(0)}} = \mathbf{u}_0$), it can be shown — at least formally — that problem (1)-(5) is *equivalent* to

For $t \geq 0$, *find* $\{\mathbf{U}(t), P(t), \lambda(t)\} \in \mathbf{V}_{\mathbf{g}_0(t)} \times L_0^2(\Omega) \times \Lambda(t)$ *such that*

$$\int_\Omega \frac{\partial \mathbf{U}}{\partial t}\cdot\mathbf{v}\, d\mathbf{x} + \nu\int_\Omega \nabla\mathbf{U}\cdot\nabla\mathbf{v}\, d\mathbf{x} + \int_\Omega (\mathbf{U}\cdot\nabla)\mathbf{U}\cdot\mathbf{v}\, d\mathbf{x} - \int_\Omega P\nabla\cdot\mathbf{v}\, d\mathbf{x}$$
$$= \int_\Omega \tilde{\mathbf{f}}\cdot\mathbf{v}\, d\mathbf{x} + \int_{\gamma(t)} \lambda\cdot\mathbf{v}\, d\gamma,\ \forall\mathbf{v} \in \mathbf{V}_0, \tag{6}$$

$$\int_\Omega q\nabla\cdot\mathbf{U}(t)\, d\mathbf{x} = 0,\ \forall q \in L^2(\Omega), \tag{7}$$

$$\int_{\gamma(t)} (\mathbf{U}(t) - \mathbf{g}_1(t))\cdot\mu\, d\gamma = 0,\ \forall\mu \in \Lambda(t), \tag{8}$$

$$\mathbf{U}(0) = \mathbf{U}_0\ in\ \Omega, \quad \mathbf{U} = \mathbf{g}_0\ on\ \Gamma, \tag{9}$$

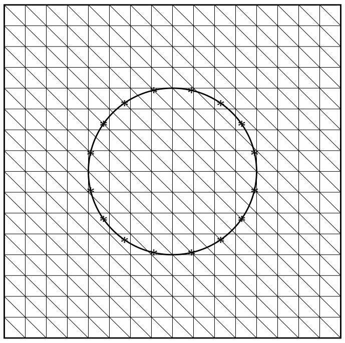
in the sense that $\mathbf{U}(t)|_{\Omega\setminus\overline{\omega(t)}} = \mathbf{u}(t)$ and $P(t)|_{\Omega\setminus\overline{\omega(t)}} = p(t)$. We can easily show that $\lambda = [\nu\partial\mathbf{U}/\partial\mathbf{n} - \mathbf{n}P]_\gamma$, where $[\ ]_\gamma$ denotes the *jump* at γ.

Remark 2.1: The mathematical analysis of flow problems such as (1)-(5) is addressed in, e.g., [AG93] (see also the references therein).

Remark 2.2: We observe that the *actual geometry*, i.e., $\omega(t)$ and $\gamma(t)$ occurs "only" in the $\gamma(t)$-integral in (6) and in (8); this is a justification of the domain embedding approach.

3 Finite Element Approximation of Problem (6)-(9)

Figure 2. Part of the triangulation of Ω with mesh points indicated by "$*$" on the disk boundary



We suppose that $\Omega \subset \boldsymbol{R}^2$ $(d = 2)$. With h a space discretization step we introduce a *finite element* triangulation $\mathcal{T}_h$ of $\overline{\Omega}$ and then $\mathcal{T}_{h/2}$ a triangulation twice finer obtained by joining the midpoints of the edges of $\mathcal{T}_h$. We define then the following finite dimensional spaces which approximate $\mathbf{V}_{\mathbf{g}_0}$, $\mathbf{V}_0$, $L^2(\Omega)$, $L^2_0(\Omega)$ respectively

$$\mathbf{V}_{\mathbf{g}_{0h}} = \{\mathbf{v}_h | \mathbf{v}_h \in C^0(\bar{\Omega})^2,\ \mathbf{v}_h|_T \in P_1 \times P_1,\ \forall T \in \mathcal{T}_h,\ \mathbf{v}_h|_\Gamma = \mathbf{g}_{0h}\}, \tag{10}$$

$$\mathbf{V}_{0h} = \{\mathbf{v}_h | \mathbf{v}_h \in C^0(\bar{\Omega})^2,\ \mathbf{v}_h|_T \in P_1 \times P_1,\ \forall T \in \mathcal{T}_h,\ \mathbf{v}_h|_\Gamma = \mathbf{0}\}, \tag{11}$$

$$L^2_h = \{q_h | q_h \in C^0(\bar{\Omega}),\ q_h|_T \in P_1,\ \forall T \in \mathcal{T}_{2h}\},\ L^2_{0h} = \{q_h | q_h \in L^2_h,\ \textstyle\int_\Omega q_h\, d\mathbf{x} = 0\}; \tag{12}$$

in (10)-(12), P_1 is the space of the polynomials in x_1, x_2 of degree ≤ 1 and $\mathbf{g}_{0h}$ is an approximation of $\mathbf{g}_0$ such that $\int_\Gamma \mathbf{g}_{0h} \cdot \mathbf{n}\, d\Gamma = 0$. Concerning the space $\Lambda_h(t)$ approximating $\Lambda(t)$, we define it by

$$\Lambda_h(t) = \{\mu_h | \mu_h \in (L^\infty(\gamma(t)))^2, \mu_h \text{ is constant on the arc joining } 2 \text{ consecutive mesh points on } \gamma(t)\}. \tag{13}$$

A particular choice for the mesh points on γ is visualized on Figure 2, where ω is a disk. Let us resist any requirement that the mesh points on γ have to be at the intersection of γ with the triangle edges of $\mathcal{T}_{h/2}$; (see [GG95] for more details and the relations between h_Ω and h_γ). This kind of decoupling between the Ω and γ meshes makes the domain embedding approach attractive for problems with moving boundaries like those discussed in this note. With the above spaces it is natural to approximate problem (6)-(9) by (with obvious notation)

$$\int_\Omega \frac{\partial \mathbf{U}_h}{\partial t} \cdot \mathbf{v}\, d\mathbf{x} + \nu \int_\Omega \nabla \mathbf{U}_h \cdot \nabla \mathbf{v}\, d\mathbf{x} + \int_\Omega (\mathbf{U}_h \cdot \nabla)\mathbf{U}_h \cdot \mathbf{v}\, d\mathbf{x} - \int_\Omega P_h \nabla \cdot \mathbf{v}\, d\mathbf{x}$$
$$= \int_\Omega \tilde{\mathbf{f}}_h \cdot \mathbf{v}\, d\mathbf{x} + \int_{\gamma(t)} \lambda_h \cdot \mathbf{v}\, d\gamma,\ \forall \mathbf{v} \in \mathbf{V}_{0h},\ \mathbf{U}_h(t) \in \mathbf{V}_{\mathbf{g}_{0(t)h}}, \tag{14}$$

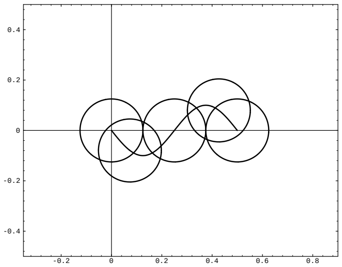
$$\int_\Omega q\nabla \cdot \mathbf{U}_h(t)\, d\mathbf{x} = 0,\ \forall q \in L_h^2,\ P_h(t) \in L_{0h}^2, \tag{15}$$

$$\int_{\gamma(t)} (\mathbf{U}_h(t) - \mathbf{g}_1(t)) \cdot \mu\, d\gamma = 0,\ \forall \mu \in \Lambda_h(t),\ \lambda_h(t) \in \Lambda_h(t), \tag{16}$$

$$\mathbf{U}_h(0) = \mathbf{U}_{0h}; \tag{17}$$

in (17), $\mathbf{U}_{0h}$ is an approximation of $\mathbf{U}_0$, approximately divergence-free.

Figure 3.



4 Time Discretization of (14)-(17) by Operator Splitting Methods

From an abstract point of view problem (14)-(17) is a particular case of the following class of initial value problems

$$\frac{d\phi}{dt} + A_1(\phi) + A_2(\phi) + A_3(\phi) = f,\ \phi(0) = \phi_0, \tag{18}$$

where the operators A_i can be *multivalued.* Among many operator splittings which can be employed to solve (18) we advocate the very simple one below (analyzed in, e.g., [Mar90]); it is only first-order accurate but its low order accuracy is compensated by good stability and robustness properties.

A fractional step scheme à la Marchuk-Yanenko: With $\triangle t$ a time discretization step and the initial guess, $\phi^0 = \phi_0$, the scheme is defined as follows:

For $n \geq 0$, we obtain ϕ^{n+1} from ϕ^n via the solution of

$$(\phi^{n+j/3} - \phi^{n+(j-1)/3})/\triangle t + A_j(\phi^{n+j/3}) = f_j^{n+1}, \tag{19}$$

with $j = 1, 2, 3$ and $\sum_{j=1}^3 f_j^{n+1} = f^{n+1}$. Applying scheme (19) to problem (14)-(17) we obtain (with $0 \leq \alpha, \beta \leq 1$, $\alpha + \beta = 1$, and after dropping some of the subscripts h):

$$\mathbf{U}^0 = \mathbf{U}_{0h}; \tag{20}$$

for $n \geq 0$, *we compute* $\{\mathbf{U}^{n+1/3}, P^{n+1/3}\}$, $\mathbf{U}^{n+2/3}$, $\{\mathbf{U}^{n+1}, \lambda^{n+1}\}$ *via the solution of*

$$\begin{cases} \displaystyle\int_\Omega \frac{\mathbf{U}^{n+1/3}-\mathbf{U}^n}{\triangle t}\cdot\mathbf{v}\,d\mathbf{x} - \int_\Omega P^{n+1/3}\nabla\cdot\mathbf{v}\,d\mathbf{x} = 0,\ \forall\mathbf{v}\in\mathbf{V}_{0h}, \\ \displaystyle\int_\Omega q\nabla\cdot\mathbf{U}^{n+1/3}\,d\mathbf{x} = 0,\ \forall q\in L_h^2;\ \mathbf{U}^{n+1/3}\in\mathbf{V}_{\mathbf{g}_{0h}}^{n+1},\ P^{n+1/3}\in L_{0h}^2, \end{cases} \tag{21}$$

$$\begin{cases} \displaystyle\int_\Omega \frac{\mathbf{U}^{n+2/3}-\mathbf{U}^{n+1/3}}{\triangle t}\cdot\mathbf{v}\,d\mathbf{x} + \alpha\nu\int_\Omega \nabla\mathbf{U}^{n+2/3}\cdot\nabla\mathbf{v}\,d\mathbf{x} \\ \displaystyle\quad + \int_\Omega(\mathbf{U}^{n+1/3}\cdot\nabla)\mathbf{U}^{n+2/3}\cdot\mathbf{v}\,d\mathbf{x} = \alpha\int_\Omega \tilde{\mathbf{f}}^{n+1}\cdot\mathbf{v}\,d\mathbf{x},\ \forall\mathbf{v}\in\mathbf{V}_{0h}; \\ \mathbf{U}^{n+2/3}\in\mathbf{V}_{\mathbf{g}_{0h}}^{n+1}, \end{cases} \tag{22}$$

$$\begin{cases} \displaystyle\int_\Omega \frac{\mathbf{U}^{n+1}-\mathbf{U}^{n+2/3}}{\triangle t}\cdot\mathbf{v}\,d\mathbf{x} + \beta\nu\int_\Omega \nabla\mathbf{U}^{n+1}\cdot\nabla\mathbf{v}\,d\mathbf{x} \\ \displaystyle\quad = \beta\int_\Omega \tilde{\mathbf{f}}^{n+1}\cdot\mathbf{v}\,d\mathbf{x} + \int_{\gamma^{n+1}}\lambda^{n+1}\cdot\mathbf{v}\,d\gamma,\ \forall\mathbf{v}\in\mathbf{V}_{0h}, \\ \displaystyle\int_{\gamma^{n+1}}(\mathbf{U}^{n+1}-\mathbf{g}_{1h}^{n+1})\cdot\mu\,d\gamma = 0,\ \forall\mu\in\Lambda_h^{n+1}; \\ \mathbf{U}^{n+1}\in\mathbf{V}_{\mathbf{g}_{0h}}^{n+1}(=\mathbf{V}_{\mathbf{g}_0((n+1)\triangle t)h}),\lambda^{n+1}\in\Lambda_h^{n+1}(=\Lambda_h((n+1)\triangle t)). \end{cases} \tag{23}$$

5 Solution of the Subproblems (21), (22) and (23)

By inspection of (21) it is clear that $\mathbf{U}^{n+1/3}$ is the $L^2(\Omega)^2$-projection of $\mathbf{U}^n$ on the (affine) subset of the functions $\mathbf{v}\in\mathbf{V}_{\mathbf{g}_{0h}}^{n+1}$ such that $\int_\Omega q\nabla\cdot\mathbf{v}\,d\mathbf{x} = 0$, $\forall q\in L_h^2$, $P^{n+1/3}$ being the corresponding Lagrange multiplier in L_{0h}^2. The pair $\{\mathbf{U}^{n+1/3}, P^{n+1/3}\}$ is *unique* and to compute it we can use an Uzawa/conjugate gradient algorithm operating in L_{0h}^2 equipped with the scalar product $\{q,q'\}\to\int_\Omega\nabla q\cdot\nabla q'\,d\mathbf{x}$. We obtain thus an algorithm preconditioned by the discrete equivalent of $-\Delta$ for the homogeneous Neumann boundary condition. Such an algorithm is *very* easy to implement and is described in [GPP96]; it seems to have excellent convergence properties.

If $\alpha > 0$, problem (22) is a classical one; it can be easily solved, for example, by a least squares/conjugate gradient algorithm like those discussed in [Glo84].

If $\beta > 0$ the solution of problem (23) has been discussed in [GPP94]. In the particular case where $\beta = 0$, problem (23) reduces to an $L^2(\Omega)^2$-projection over the subspace of $\mathbf{V}_{\mathbf{g}_{0h}}^{n+1}$ of the functions $\mathbf{v}$ satisfying the condition $\int_{\gamma^{n+1}}(\mathbf{v}-\mathbf{g}_{1h}^{n+1})\cdot\mu\,d\gamma = 0$, $\forall\mu\in\Lambda_h^{n+1}$. It follows from the above observation that if $\beta = 0$, problem (23) can be solved by an Uzawa/conjugate gradient algorithm operating in Λ_h^{n+1}, which has many similarities with the algorithm used to solve problem (21). If one uses the trapezoidal rule to compute the various $L^2(\Omega)$-integrals in (23), taking $\beta = 0$ brings further simplification since in that particular case $\mathbf{U}^{n+1}$ will coincide with $\mathbf{U}^{n+2/3}$ at those vertices of $\mathcal{T}_{h/2}$ such that the support of the related shape function does not intersect γ^{n+1}; from the above observation it follows that to obtain $\mathbf{U}^{n+1}$ and λ^{n+1} we have to solve a linear system of the following form

$$A\mathbf{x} + B^t\mathbf{y} = \mathbf{b},\ \ B\mathbf{x} = \mathbf{c}. \tag{24}$$

For the numerical simulations presented in Section 6 we have used $\alpha = 1$ and $\beta = 0$ in (22), (23).

6 Numerical Experiments

We simulate a two-dimensional flow with $\Omega = (-0.35, 0.9) \times (-0.5, 0.5)$ (see Figure 3) and ω a moving disk of radius 0.125. The center of the disk is moving between $(0,0)$ and and $(0.5,0)$ along a prescribed trajectory $(x(t), y(t)) = (0.25(1 - \cos(\frac{\pi t}{2})), -0.1\sin(\pi(1-\cos(\frac{\pi t}{2}))))$ (see Figure 3) of period 4. Several different positions of the disk have been shown on Figure 3. The boundary conditions are $\mathbf{u} = \mathbf{0}$ on Γ and $\mathbf{u}$ on $\partial\omega(t)$ coinciding with the disk velocity. We suppose that the disk rotates counterclockwise at angular velocity 2π. Since we are taking $\nu = 0.005$, the maximum Reynolds number based on the disk diameter as characteristic length is 102.336. On Ω we have used a regular triangulation $\mathcal{T}_{h/2}$ to approximate the velocity, like the one in Figure 2, the pressure grid $\mathcal{T}_h$ being twice coarser. Concerning $\Lambda_h(t)$, $\gamma(t)$ has been divided into M subarcs of equal length. We have done two simulations: For the first one we have taken $h = 1/128$, $\triangle t = 0.00125$ and $M = 80$. For the second we have taken $h = 1/256$, $\triangle t = 0.00125$ and $M = 160$. With stopping criteria of the order of 10^{-12} we need around 10 iterations at most to have convergence of the conjugate gradient algorithms used to solve the problems at each step of the scheme (20)-(23). On Figure 4, we show the isobar lines, the vorticity density and the streamlines obtained at t =5, 6, 7, 8 for $h = 1/256$, $\triangle t = 0.00125$ and $M = 160$. There is a good agreement between the results obtained from these two simulations.

Acknowledgement

We would like to acknowledge the helpful comments and suggestions of E. J. Dean, J. W. He, H. H. Hu, D. D. Joseph, Y. Kuznetsov, B. Maury, A. H. Sameh and F. J. Sanchez, and the support of University of Houston and the Department of Computer Science, University of Minnesota. We also benefited from the support of NSF (Grants DMS 8822522, DMS 9112847, DMS 9217374), DRET (Grant 89424), DARPA (Contracts AFOSR F49620-89-C-0125 and AFOSR-90-0334), the Texas Board of Higher Education (Grants 003652156ARP and 003652146ATP), University of Houston (PEER grant 1-27682) and again NSF under the HPCC Grand Challenge Grant ECS-9527123.

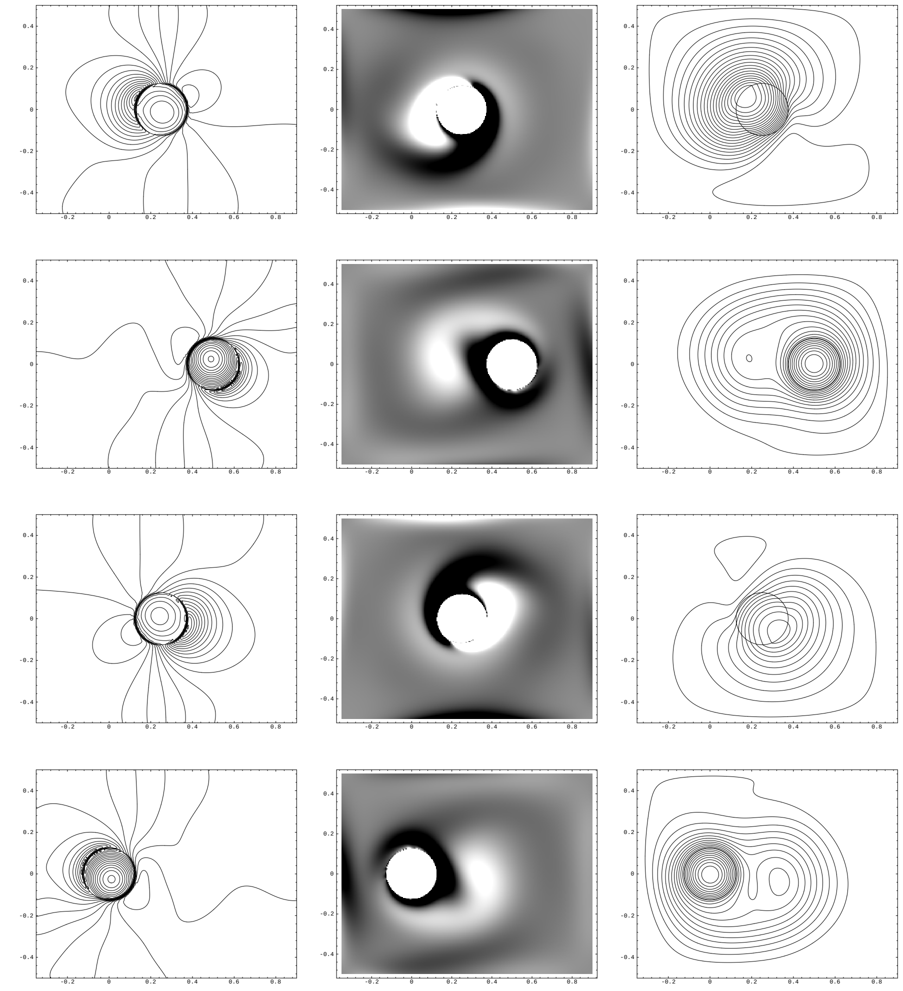
REFERENCES

[AG93] Amiez G. and Gremaud P. A. (1993) On a penalty method for the navier-stokes problem in regions with moving boundaries. *Comp. Appl. Math.* 12: 113–122.

[GG95] Girault V. and Glowinski R. (1995) Error analysis of a fictitious domain method applied to a dirichlet problem. *Japan J. of Industrial and Applied Mathematics* 12: 487–514.

[Glo84] Glowinski R. (1984) *Numerical Methods for Nonlinear Variational Problems.* Springer-Verlag, New York.

Figure 4. Isobar lines (at left), vorticity density (at middle) and streamlines (at right) at time t =5, 6, 7, 8 in one period of disk motion. The disk moves from the left to the right, then to the left. The mesh size for velocity (resp., pressure) is $h = 1/256$ (resp., $h = 1/128$).



[GPP94] Glowinski R., Pan T. W., and Periaux J. (1994) A fictitious domain method for dirichlet problem and applications. *Comp. Meth. Appl. Mech. Eng.* 111: 283–303.

[GPP96] Glowinski R., Pan T. W., and Periaux J. (1996) Fictitious domain methods for incompressible viscous flow around moving rigid bodies. In *Proceedings of MAFELAP 1996 (to appear)*.

[Mar90] Marchuk G. I. (1990) Splitting and alternate direction methods. In Ciarlet P. and Lions J. (eds) *Handbook of Numerical Analysis*. Vol. I, North-Holland, Amsterdam.

[Pes72] Peskin C. Y. (1972) Flow patterns around heart valves: A numerical method. *J. Comp. Phys.* 10: 252–271.

41

An Efficient FGMRES Solver for the Shallow Water Equations based on Domain Decomposition

Serge Goossens, Kian Tan, and Dirk Roose

1 Shallow Water Equations

The *Shallow Water Equations* (SWE) are a set of nonlinear hyperbolic equations, describing long waves relative to the water depth. Physical phenomena such as tidal waves in rivers and seas, breaking of waves on shallow beaches and even harbour oscillations can be modelled successfully with the SWE. The 3D SWE (1.1)–(1.3) given below for Cartesian (ξ, η) coordinates are based on the *hydrostatic assumption*, that the influence of the vertical component of the acceleration of the water particles on the pressure can be neglected.

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial \xi} + v\frac{\partial u}{\partial \eta} + \frac{\omega}{H}\frac{\partial u}{\partial \sigma} - fv + g\frac{\partial \zeta}{\partial \xi} - \nu_H \left(\frac{\partial^2 u}{\partial \xi^2} + \frac{\partial^2 u}{\partial \eta^2}\right) - \frac{1}{H^2}\frac{\partial}{\partial \sigma}\left(\nu_V \frac{\partial u}{\partial \sigma}\right) = 0 \tag{1.1}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial \xi} + v\frac{\partial v}{\partial \eta} + \frac{\omega}{H}\frac{\partial v}{\partial \sigma} + fu + g\frac{\partial \zeta}{\partial \eta} - \nu_H \left(\frac{\partial^2 v}{\partial \xi^2} + \frac{\partial^2 v}{\partial \eta^2}\right) - \frac{1}{H^2}\frac{\partial}{\partial \sigma}\left(\nu_V \frac{\partial v}{\partial \sigma}\right) = 0 \tag{1.2}$$

$$\frac{\partial \zeta}{\partial t} + \frac{\partial (Hu)}{\partial \xi} + \frac{\partial (Hv)}{\partial \eta} + \frac{\partial \omega}{\partial \sigma} = 0 \tag{1.3}$$

We denote by ζ the water elevation above some plane of reference, hence the total water depth is given by $H = d + \zeta$, where d is the depth below this plane of reference. The scaled vertical coordinate $\sigma = \frac{z - \zeta}{d + \zeta}$ varies between -1 at the bottom and 0 at the free surface. The velocities in the ξ- and η-directions are denoted by u and v respectively, while ω represents the transformed vertical velocity. The parameter f accounts for the Coriolis force due to the rotation of the Earth. The viscosity is modelled using ν_H and ν_V. In each σ-plane ν_H models the "horizontal" viscosity, while ν_V describes the viscosity in the vertical (σ) direction.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 Alternating Operator Implicit Method

For the time integration we use the two-stage Alternating Operator Implicit (AOI) time splitting method, which has been developed at Delft Hydraulics [dG93]. This method is unconditionally stable and second order accurate in time. In the first stage (most of) the advection and diffusion terms in the momentum equations are handled implicitly, while the continuity equation is integrated explicitly. The resulting two linear systems for the intermediate u and v are solved by Red-Black Gauss-Seidel iterations. During the second stage the continuity equation is treated implicitly. Substitution into the continuity equation of the momentum equations, in which the velocity components are now handled explicitly, leads to a nonlinear system for the water elevation ζ. For each time step n, we perform Q fixed point iterations to solve this nonlinear system. Introducing an iteration counter q ($q = 1, 2, \dots, Q$) and multiplying the pressure terms in the momentum equations with $H^{(n,q)}/H^{(n,q+1)}$, we obtain

$$\left(I - \nu_\xi \frac{\partial^2}{\partial \xi^2} - \nu_\eta \frac{\partial^2}{\partial \eta^2} \right) \zeta^{(n,q)} = f^*, \tag{2.4}$$

where the right-hand side f^* involves previously computed values and where $\zeta^{(n,q)}$ denotes the water elevation at iteration q of time step n. In the remainder of the paper we drop the superscripts. The imposed boundary conditions might be of Neumann type (e.g. closed wall) which could lead to a nonsymmetric linear system after discretisation. The pseudo viscosities ν_ξ and ν_η mainly depend on the time step and the total depth, which makes the linear system *nonsymmetric.*

Since the classical five point star stencil is used, a discrete equation of the form

$$(b_{i,j} + b_{i,j}^{(x)} + b_{i,j}^{(y)})\zeta_{i,j} + a_{i,j}\zeta_{i-1,j} + c_{i,j}\zeta_{i+1,j} + d_{i,j}\zeta_{i,j-1} + e_{i,j}\zeta_{i,j+1} = f_{i,j} \tag{2.5}$$

is obtained for each grid point (i, j) and the resulting linear system has a pentadiagonal structure. In practice it often suffices to take $Q = 2$. Until recently an ADI iteration was used for solving system (2.4).

The main topic addressed in this paper is the application of a *Domain Decomposition Preconditioner* in combination with the *Flexible GMRES* (FGMRES) method to solve (2.4). The original ADI method is used as a preconditioner in the subdomain solver only.

3 Generalised Additive Schwarz Preconditioner

The domain decomposition preconditioner which is employed in accelerating the FGMRES method to solve (2.4) on the entire domain is based on a *Generalised Additive Schwarz Preconditioner* (GASP). Let $R_i : \Omega \mapsto \Omega_i$ denote the (linear) restriction operator that maps onto subdomain i by selecting the components corresponding to this subdomain. The matrix $M_i = R_i A R_i^T$ denotes the principal submatrix of the matrix A associated with subdomain Ω_i. The result of applying the GASP can be written as a sum of the extensions of the solutions of independent subdomain problems, which can be solved in parallel.

$$M^{-1} = \sum_{i=1}^{p} R_i^T M_i^{-1} R_i \tag{3.6}$$

We elaborate on this GASP for the case of two subdomains separated by the interface Γ as shown in Fig. 1. Extension to more subdomains is straightforward. At the heart of our GASP lies an *extension* of the subdomains to (physically) slightly overlapping grids. With a proper definition of the overlap, the restrictions R_i can be defined in such a way that the original discretisation is "distributed" across the subdomain operators M_i. Since the classical five point star stencil is used an overlap of two grid lines is sufficient. Figure 2 illustrates the extension process. In the discretisation, points in subdomain Ω_1 are only connected to points in Ω_1 or in Ω_l. Similar statements can be made about the points in Ω_l, Ω_r and Ω_2. This leads to the following block structured linear system.

Figure 1 Grid before partitioning

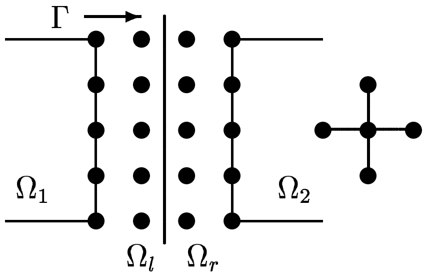
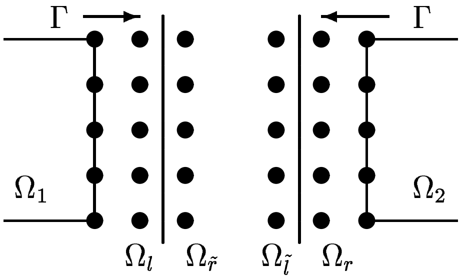


Figure 2 Grid after partitioning



$$\begin{pmatrix} A_{11} & A_{1l} & 0 & 0 \\ A_{l1} & A_{ll} & A_{lr} & 0 \\ 0 & A_{rl} & A_{rr} & A_{r2} \\ 0 & 0 & A_{2r} & A_{22} \end{pmatrix} \begin{pmatrix} \zeta_1 \\ \zeta_l \\ \zeta_r \\ \zeta_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_l \\ f_r \\ f_2 \end{pmatrix} \tag{3.7}$$

After extension towards overlap, and thus duplication of Ω_l and Ω_r into $\Omega_{\tilde{l}}$ and $\Omega_{\tilde{r}}$, we obtain an enhanced system of equations in which we still have to specify the relation between the "overlapping" unknowns. The obvious way is just to state that the values in the duplicated subdomains $\Omega_{\tilde{l}}$ and $\Omega_{\tilde{r}}$ should be copied from the values in the original subdomains Ω_l and Ω_r respectively. This is known as the *Dirichlet-Dirichlet* (DD) coupling. The enhanced system of equations with this DD coupling can be written as follows.

$$\begin{pmatrix} A_{11} & A_{1l} & 0 & 0 & 0 & 0 \\ A_{l1} & A_{ll} & A_{lr} & 0 & 0 & 0 \\ 0 & 0 & I & 0 & -I & 0 \\ 0 & -I & 0 & I & 0 & 0 \\ 0 & 0 & 0 & A_{rl} & A_{rr} & A_{r2} \\ 0 & 0 & 0 & 0 & A_{2r} & A_{22} \end{pmatrix} \begin{pmatrix} \zeta_1 \\ \zeta_l \\ \tilde{\zeta}_r \\ \tilde{\zeta}_l \\ \zeta_r \\ \zeta_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_l \\ 0 \\ 0 \\ f_r \\ f_2 \end{pmatrix} \tag{3.8}$$

Tan [TD88] showed that the spectral radius of the preconditioned operator AM^{-1} and thus the convergence properties of a *Krylov Subspace Method* preconditioned by a GASP as given by (3.6), are improved by pre-multiplying the linear system with a properly chosen nonsingular matrix P of the form.

$$P = \begin{pmatrix} I & 0 & 0 & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 & 0 \\ 0 & 0 & C_{lr} & -C_{ll} & 0 & 0 \\ 0 & 0 & -C_{rr} & C_{rl} & 0 & 0 \\ 0 & 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & 0 & I \end{pmatrix} \tag{3.9}$$

This can also be interpreted in terms of imposing more general conditions at the subdomain interfaces. This approach was originally introduced by Lions [Lio90] and subsequently used by e.g. Hagstrom et. al. [HTJ88] and Nataf and Rogier [NR95]. The submatrices C_{lr}, C_{ll}, C_{rr} and C_{rl} are chosen to achieve a clustering of the eigenvalues of the preconditioned operator, subject to the condition that P remains nonsingular. This gives rise to the *Locally Optimised Block Jacobi* preconditioners which are thus based on the enhanced system of equations $A\zeta = f$:

$$\begin{pmatrix} A_{11} & A_{1l} & 0 & 0 & 0 & 0 \\ A_{l1} & A_{ll} & A_{lr} & 0 & 0 & 0 \\ 0 & C_{ll} & C_{lr} & -C_{ll} & -C_{lr} & 0 \\ 0 & -C_{rl} & -C_{rr} & C_{rl} & C_{rr} & 0 \\ 0 & 0 & 0 & A_{rl} & A_{rr} & A_{r2} \\ 0 & 0 & 0 & 0 & A_{2r} & A_{22} \end{pmatrix} \begin{pmatrix} \zeta_1 \\ \zeta_l \\ \tilde{\zeta}_r \\ \tilde{\zeta}_l \\ \zeta_r \\ \zeta_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_l \\ 0 \\ 0 \\ f_r \\ f_2 \end{pmatrix} \tag{3.10}$$

This enhanced system of equations can be written in terms of the 3×3 blocks. Defining the restriction operators R_1 and R_2 in terms of the index sets corresponding to ζ_1, ζ_l and $\tilde{\zeta}_r$ on the one hand and $\tilde{\zeta}_l$, ζ_r and ζ_2 on the other hand, the GASP can be written as the block diagonal matrix M with

$$M = \begin{pmatrix} R_1 A R_1^T & 0 \\ 0 & R_2 A R_2^T \end{pmatrix}. \tag{3.11}$$

4 Flexible GMRES

Applying FGMRES in combination with the GASP described above to solve (2.4) is straightforward. The FGMRES method developed by Saad [Saa93] is a Krylov subspace method which allows the introduction of a set of well-chosen vectors in the search space. We assume for convenience that AM^{-1} is normal. The FGMRES algorithm computes the fundamental relation

$$AZ_m = V_m H_m + h_{m+1,m} v_{m+1} e_m^T \tag{4.12}$$

where $Z_m = \begin{pmatrix} z_1 & z_2 & \dots & z_m \end{pmatrix}$ is the matrix containing the search directions and the matrix $V_m = \begin{pmatrix} v_1 & v_2 & \dots & v_m \end{pmatrix}$ is defined by its columns. The matrix H_m is a square $m \times m$ upper Hessenberg matrix whose elements are computed during the orthogonalisation process of the v-vectors, consequently $V_m^T V_m = I$. Using a fixed preconditioner with FGMRES is equivalent to using right preconditioned GMRES with this preconditioner. In this case, the matrix Z_m can be computed by applying the fixed preconditioning operator M^{-1} to the matrix V_m.

$$Z_m = M^{-1} V_m \tag{4.13}$$

The GASP described above is a fixed preconditioner if and only if the linear systems in the subdomains are solved to full precision. The convergence of FGMRES is mainly governed by the eigenvalue distribution of the preconditioned operator AM^{-1}. In particular, convergence acceleration can be expected for well-separated extreme eigenvalues. Also, the stage of the FGMRES process in which acceleration occurs is related to the convergence of the Ritz values to extreme eigenvalues [VdSVdV86, VdVV93]. This phenomenon can be made visible by explicitly computing the Ritz values, i.e. the eigenvalues of $H_m = V_m^T(AM^{-1})V_m$, in the course of the FGMRES process. The eigenvectors corresponding to the outliers of this spectrum represent the eigenvector components to be removed from the initial residual which "uphold" the convergence. Due to the construction of the GASP considered here, this Ritz spectrum, at least for meshes not too fine, typically resembles the spectrum as depicted in Fig. 4, i.e. a few well-separated outliers and a cluster around 1. For comparison we show in Fig. 3 the Ritz spectrum of the domain decomposition preconditioner for the same problem when Dirichlet-Dirichlet coupling is used. This spectrum does not show a clear separation of a cluster of eigenvalues around 1 and some outliers. On the contrary, the eigenvalues are spread out over the open interval $(0, 2)$ and a lot of the eigenvalues are either close to 0 or to 2. The eigenvalue distribution explains the slow convergence of this domain decomposition method with DD coupling when it is used as a solver, because the spectral radius of the matrix $(I - AM^{-1})$ is close to 1. In the next section we try to exploit the nice spectral properties of the GASP.

Figure 3 Spectrum of H_m: Dirichlet-Dirichlet Coupling

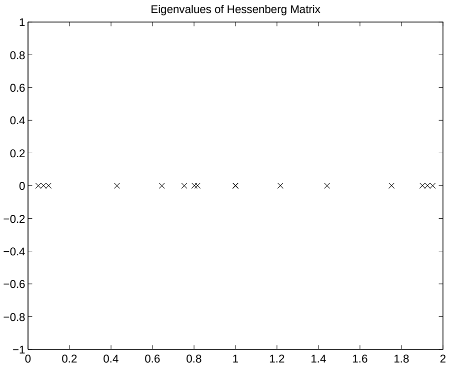


Figure 4 Spectrum of H_m: Locally Optimised Coupling

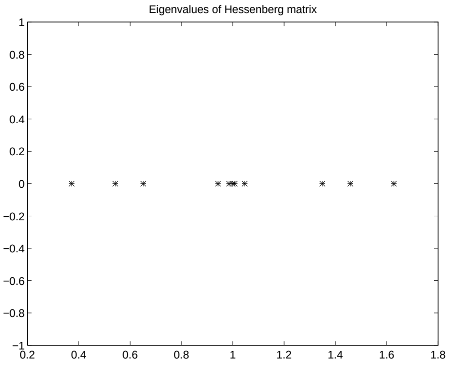


Table 1 Number of iterations needed to solve the second linear system when the reuse of vectors in the subspace is done by truncation, assembling (rank-k) or assembling of preconditioned Ritz vectors (k outliers) for the rectangular basin partitioned in 4 strips.

truncation	GASP appl.	rank-k	GASP appl.	k outliers	GASP appl.
z_1, z_2	9 or 10	2	9	2	8
z_1, z_2, z_3	9 or 10	3	8 or 9	3	7 or 8
$z_1, \ldots, z_4$	8 or 9	4	8	4	7
$z_1, \ldots, z_5$	8	5	7	5	6
$z_1, \ldots, z_6$	7 or 8	6	6	6	5
$z_1, \ldots, z_7$	6 or 7	7	6		
$z_1, \ldots, z_8$	6	8	6		
$z_1, \ldots, z_9$	5 or 6	9	5 or 6		
$z_1, \ldots, z_{10}$	5	10	5 or 6		

5 Reuse strategies

The main motivation for using FGMRES instead of GMRES is that the former — in contrast to the latter — accommodates variable preconditioning; any vector z can be put into the search space Z_m as long as its image Az is known in order to be able to compute the correction to the residual. This property in combination with the observation that our specific time integration method results for each time step in a sequence of systems (2.4) has raised the question whether it is possible to reuse previously computed search vectors during the solution of the next systems by FGMRES. Obviously, one advantage of reusing vectors is that it is a lot cheaper than applying the (expensive) GASP which after all requires the solution of a linear system in each subdomain. Also, when (approximations of) the preconditioned eigenvectors that uphold the convergence are collected in the search space, accelerated convergence might be achieved from the first newly computed z-vector on. Several strategies to reuse vectors from an already generated subspace have been tested. In practice, we always use $Q = 2$, i.e. two systems must be solved in each time step. We focus on the solution of the second system in each time step ($q = 2$), possibly reusing information from the search space Z_m built during the solution of the first system ($q = 1$). We formulate the following reuse strategies:

1. **truncation:** introduce the first k z-vectors $z_i^{(R)} = z_i$ $(i = 1, \ldots, k)$.
2. **assembling**: introduce the best rank-k approximation of $\text{SPAN}\{z_1, z_2, \ldots, z_m\}$.
3. **assembling of preconditioned Ritz vectors:** introduce k preconditioned approximate eigenvectors corresponding to outliers: $z_i^{(R)} = Z_m y_i$ $(i = 1, \ldots, k)$, where y_i is the eigenvector of H_m corresponding to the eigenvalue λ_i, i.e. $H_m y_i = \lambda y_i$.

The trivial truncation strategy 1 gives an indication of what can be expected from more sophisticated reuse strategies. The case in which all z-vectors are reused, allows us to verify whether the Arnoldi process is able to quickly generate a reasonably approximate eigenspectrum. The second strategy requires the computation of the singular value decomposition of $Z_m = U\Sigma V^T$. When the singular values are ordered $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_m \geq 0$, the best rank-$k$ approximation of $\textsc{span}\{z_1, z_2, \ldots, z_m\}$ is given by $\textsc{span}\{u_1, u_2, \ldots, u_k\}$. The use of a best rank-k approximation is motivated by the fact that the column space of Z_m contains preconditioned approximate eigenvectors corresponding to the outliers. The hope is that a lower dimensional approximation still contains an approximation of these eigenvectors. The results seem to indicate that this strategy is not entirely capable of filtering out the preconditioned eigenspace. The third reuse strategy relies on the observation of clustered eigenspectra in combination with a (small) number of clearly distinguishable outliers as is the case in Fig. 4. An explicit construction of the preconditioned eigenspace corresponding to the outliers is then possible. Note that the construction is also based on (4.13). Based on the results in Table 1 the assembling strategy of preconditioned Ritz vectors has been chosen for further experiments.

6 Test Case and Results

The test case is concerned with the flow in a 8000m by 1200m rectangular basin which is 8m deep. The uniform grid has one layer in the vertical (σ) direction which contains 80×12 grid points in the horizontal direction. The prescribed boundary conditions are as follows. The north and south boundaries are closed, leading to Neumann boundary conditions in (2.4). At the east boundary the water elevation is kept constant at $\zeta = 2$m. At the west boundary the water elevation is prescribed to model the tide, yielding $\zeta = 2 + \sin \frac{2\pi t}{3600}$ m. Tests have been carried out with stripwise partitionings of the rectangular domain into 4 and 8 subdomains. A test with 4 subdomains and a mesh width of 50m instead of 100m was also done to see the effect of refinement on the number of outliers and the separation between the outliers and the cluster.

The convergence histories showing the scaled (with $\|f\|_2$) residual norm as a function of the dimension of the search space for the FGMRES algorithm applied to the two linear systems arising each time step are shown in Fig. 5. The convergence history for the first linear system starts at about 0.2 and drops below the adopted threshold of 10^{-9} after 11 iterations. This requires 11 applications of the GASP. Figure 4 shows the eigenvalues of the Hessenberg matrix H_m constructed by FGMRES during the solution of this first linear system. The six eigenvalues that are not close to 1 are the outliers which "hamper" fast convergence of FGMRES. The convergence history for the second linear system starts off with a plateau at about 0.001, dropping sharply to reach the tolerance criterion after 11 iterations as well. The plateau corresponds to the reuse of assembled preconditioned Ritz vectors. This corresponds to the removal of the approximate eigenvectors associated with the outliers from the initial residual, a process which hardly reduces the norm of it. However it makes FGMRES converge as if the outliers were not present at all; starting from the first newly computed search vector the residuals decrease rapidly, at the same speed as in the end stage of the solution of the first system. Because of the reuse, solving the second linear system requires only 5 applications of the GASP.

Instead of computing the approximate eigenvectors at each time step from the matrix Z_m constructed during the solution of the first linear system ($q = 1$), we

Figure 5 Convergence histories of the preconditioned FGMRES method

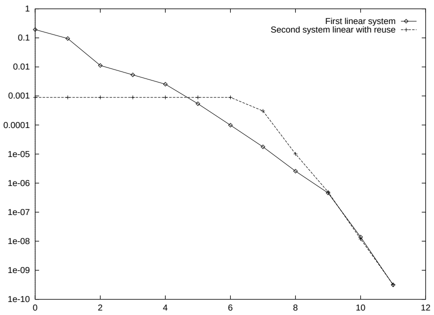
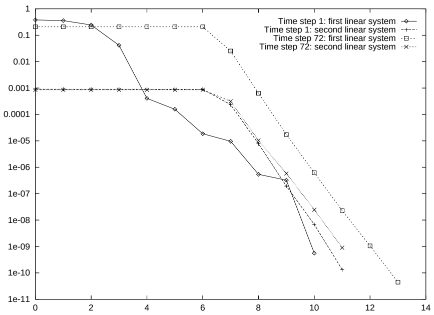


Figure 6 Convergence histories of the preconditioned FGMRES method



can also construct the approximate eigenspace only once, i.e. from the first linear system arising in the first time step. Moreover, the eigenspace is now reused in the solution process for the first linear systems ($q = 1$) of each time step as well. The convergence histories are shown in Fig. 6. As can be seen from this Fig. we save on preconditioning steps by introducing these vectors also in the search space when solving the first linear system. The convergence histories for the second linear system show that it is not necessary to compute the approximate eigenvectors at each time step, since the results with the vectors from the first time step are sufficiently close to those with the vectors from the current time step.

7 Conclusion

We have developed a Generalised Additive Schwarz Preconditioner for use within FGMRES to solve linear systems arising in the solution of the time-dependent shallow water equations. The preconditioned operator AM^{-1} has a clustered eigenspectrum with only a few outlying eigenvalues, at least for meshes not too fine. This property together with the specific time integration method enables the reuse of search vectors in the FGMRES process which leads to reductions in computation time.

Acknowledgement

This paper presents results obtained during the stay of S. Goossens at Delft Hydraulics. The authors thank E. de Goede for his help and also acknowledge the financial support by the E.C. MAST II Concerted Action “Application of High Performance Computing Techniques for the Modeling of Marine Ecosystems”.

REFERENCES

[dG93] de Goede E. D. (September 1993) Een AOI methode voor TRISULA. Technical Report VR595.93/Z642, Delft Hydraulics. in dutch.

[HTJ88] Hagstrom T., Tewarson R. P., and Jazcilevich A. (1988) Numerical experiments on a domain decomposition algorithm for nonlinear elliptic boundary value problems. *Appl. Math. Lett.* 1(3): 299–302.

[Lio90] Lions P. L. (1990) On the Schwarz alternating method III: A variant for nonoverlapping subdomains. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Proc. Third Int. Conf. on Domain Decomposition Meths.*, pages 202–223. SIAM, Philadelphia.

[NR95] Nataf F. and Rogier F. (1995) Factorization of the convection-diffusion operator and the Schwarz algorithm. *Mathematical Models and Methods in Applied Sciences* 5(1): 67–93.

[Saa93] Saad Y. (1993) A flexible inner-outer preconditioned GMRES algorithm. *SIAM J. Sci. Comput.* 14(2): 461–469.

[Tan95] Tan K. H. (1995) *Local Coupling in Domain Decomposition.* PhD thesis, Universiteit Utrecht.

[VdSVdV86] Van der Sluis A. and Van der Vorst H. A. (1986) The rate of convergence of conjugate gradients. *Numerische Mathematik* 48: 543–560.

[VdVV93] Van der Vorst H. A. and Vuik C. (1993) The superlinear convergence behaviour of GMRES. *Journal of Computational and Applied Mathematics* 48: 327–341.

42

Multilevel Extension Techniques in Domain Decomposition Preconditioners

Gundolf Haase

One component in Additive Schwarz Method (ASM) Domain Decomposition (DD) preconditioners [BPS89, SBG96] using inexact subdomain solvers [Boe89, HLM91] consists of an operator extending the boundary data into the interior of each subdomain, i.e., a homogeneous extension with respect to the differential operator given in that subdomain. This paper is concerned with the construction of cheap extension operators using multilevel nodal bases [Yse86, Xu89, BPX90, Osw94] from an implementation viewpoint. Additional smoothing sweeps in the extension operators further improve the condition number of the preconditioned system. The paper summarizes and improves results given in [HLMN94, Nep95, Haa97].

1 The ASM-DD-Preconditioner

Consider the symmetric, $\mathbb{V}_0 = \overset{\circ}{\mathbb{H}}{}^1(\Omega)$–elliptic and $\mathbb{V}_0$–bounded variational problem

$$\text{find } u \in \mathbb{V}_0 : \int_\Omega \lambda(x)\, \nabla^T u(x)\, \nabla v(x)\, dx = \int_\Omega f(x)\, v(x)\, dx \qquad \forall v \in \mathbb{V}_0\,, \tag{1.1}$$

arising from the weak formulation of a scalar second–order, symmetric and uniformly bounded elliptic boundary value problem given in a plane bounded domain $\Omega \subset \mathbb{R}^2$ with a piecewise smooth boundary $\Gamma = \partial\Omega$. The material coefficients $\lambda(x) \geq \lambda_0 > 0 \quad \forall x \in \overline{\Omega}$ have to be restricted for certain extension techniques.

The domain Ω will be decomposed into p non-overlapping subdomains Ω_i $(i = 1, \ldots, p)$ such that $\overline{\Omega} = \bigcup_{i=1}^{p} \overline{\Omega}_i$. The discretization process using Courant's linear triangular finite elements in each subdomain Ω_i results in a conforming triangulation of Ω . In the following, the indices "C" and "I" correspond to nodes belonging to the coupling boundaries $\Gamma_C = \bigcup_{i=1}^{p} \partial\Omega_i \setminus \Gamma_D$ and to the interior $\Omega_I = \bigcup_{i=1}^{p} \Omega_i$ of the subdomains, respectively. Γ_D is that part of $\partial\Omega$ where Dirichlet–type boundary conditions are given whereas Neumann boundary conditions will be handled as coupling boundaries.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Define the usual finite elements (FE) nodal basis

$$\Phi = [\Phi_C, \Phi_I] = \left[\psi_1, \cdots, \psi_{N_C}, \psi_{N_C+1}, \cdots, \psi_{N_C+N_{I,1}}, \cdots, \psi_{N=N_C+N_I}\right] , \tag{1.2}$$

where the first N_C basis functions belong to Γ_C, the next $N_{I,1}$ to Ω_1, the next $N_{I,2}$ to Ω_2 and so on such that $N_I = \sum_{i=1}^{p} N_{I,i}$. Then the FE isomorphism leads to the symmetric and positive definite system of equations

$$K\,\underline{u} := \begin{pmatrix} K_C & K_{CI} \\ K_{IC} & K_I \end{pmatrix} \begin{pmatrix} \underline{u}_C \\ \underline{u}_I \end{pmatrix} = \begin{pmatrix} \underline{f}_C \\ \underline{f}_I \end{pmatrix} =: \underline{f} , \tag{1.3}$$

where $K_I = \text{blockdiag}\,(K_{I,i})_{i=1,\dots,p}$ is block diagonal and symmetric, positive definite.

Solving system (1.3) with some parallelized iterative method, e.g., CG-method, we use the ASM–DD preconditioner

$$C = \begin{pmatrix} I_C & -B_{IC}^{-T} \\ O & I_I \end{pmatrix} \begin{pmatrix} C_C & O \\ O & C_I \end{pmatrix} \begin{pmatrix} I_C & O \\ -B_{IC} & I_I \end{pmatrix} . \tag{1.4}$$

This preconditioner contains the three components C_C, $C_I = \text{diag}\,(C_{I,i})_{i=1,\dots,p}$ and $B_{IC} = \text{blockmatrix}\,(B_{IC,i})_{i=1,\dots,p}$, which can freely be chosen in order to adapt the preconditioner to the particulars of the problem under consideration. For the choice $B_{IC,i} = -B_{I,i}K_{IC,i}$ see [HLM91]. As preconditioner C_C for the Schur complement $S_C = K_C - K_{CI}K_I^{-1}K_{IC}$ the BPS [BPS89] and the S(chur)-BPX [TCK92] are used.

The preconditioning step $\underline{w} = C^{-1}\underline{r}$ can be rewritten in the form

Algorithm 1 : The ASM-DD Preconditioner [HLM91]

$$\begin{aligned} \underline{\mathbf{w}}_C &= C_C^{-1} \sum_{i=1}^{p} A_{C,i}^T \left(\underline{r}_{C,i} + B_{IC,i}^T \underline{r}_{I,i}\right) \\ \underline{\mathbf{w}}_{I,i} &= C_{I,i}^{-1} \underline{r}_{I,i} + B_{IC,i} \underline{\mathbf{w}}_{C,i} \qquad ; \ i = 1, 2, \dots, p \end{aligned}$$

where $A_i = \left(\begin{smallmatrix} A_{C,i} & A_{CI,i} \\ A_{IC,i} & A_{I,i} \end{smallmatrix}\right)$ denotes the subdomain connectivity matrix which is used for a convenient notation only. The subdomain FE assembly process which is connected with nearest neighbour communication stands behind this notation. Other DD-preconditioners and modifications of Algorithm 1 can be found in [HL92].

Assume positive, h-independent spectral equivalence constants $\underline{\gamma}_C, \overline{\gamma}_C, \underline{\gamma}_I, \overline{\gamma}_I$ fulfilling the spectral equivalence inequalities

$$\underline{\gamma}_C\, C_C \le S_C \le \overline{\gamma}_C\, C_C \qquad \text{and} \qquad \underline{\gamma}_I\, C_I \le K_I \le \overline{\gamma}_I\, C_I . \tag{1.5}$$

If we have a constant c_E so that

$$\left\| \begin{pmatrix} \underline{v}_C \\ B_{IC}\underline{v}_C \end{pmatrix} \right\|_K \le c_E \, \| \, \underline{v}_C \, \|_{S_C} \qquad \forall \underline{v}_C \in \mathbb{R}^{N_c} \tag{1.6}$$

holds then the upper and lower bounds of the condition number $\kappa(C^{-1}K)$ [HLM91, Che93] can be estimated as

$$\mathcal{O}(c_E^2) \le \kappa(C^{-1}K) \le \mathcal{O}(c_E^4) . \tag{1.7}$$

In the remaining chapter we construct extension techniques defining $B_{IC,i}$ which are cheap to implement and result in a constant c_E independent of or slightly dependent on the discretization parameter h and the number of levels ℓ.

In the following the subscript i denoting the subdomain number will be omitted.

2 Multilevel Extension Operators

Let space $\mathbb{W}_k$ consist of real-valued functions which are continuous on Ω and linear on the triangles in Ω_k^h. The space $\mathbb{V}_k$ is the space of traces on Γ of functions from $\mathbb{W}_k$.

The multilevel extension of a FE function $'^h \in \mathbb{V}_\ell$ on the boundary $\partial\Omega$ into a function $u^h \in \mathbb{W}_\ell$ in the interior of the domain Ω via the extension operator $\mathcal{E}_{\overline{\Omega}\Gamma}$ consists of three steps:

1. Choose the projection $\mathcal{Q}_k$ from $\mathbb{V}_\ell$ into $\mathbb{V}_k$ $(k = 0, \dots, \ell)$.
2. Split the function $'^h$ into a multilevel nodal basis according to the projections $\mathcal{Q}_k$

$$\chi_0^h = \mathcal{Q}_0 {'^h}, \tag{2.8a}$$

$$\chi_k^h = (\mathcal{Q}_k - \mathcal{Q}_{k-1}){'^h} \qquad k = 1, \dots, \ell\,. \tag{2.8b}$$

3. Define the extensions $u_k^h \in \mathbb{W}_k \setminus \mathbb{W}_{k-1}$ of the function χ_k^h

$$u_0^h(x_i^{(0)}) = \begin{cases} \chi_0^h(x_i^{(0)}) & , x_i^{(0)} \in \Gamma, \\ \overline{\chi} & , x_i^{(0)} \notin \Gamma, \end{cases} \tag{2.9a}$$

$$u_k^h(x_i^{(k)}) = \begin{cases} \chi_k^h(x_i^{(k)}) & , x_i^{(k)} \in \Gamma, \\ 0 & , x_i^{(k)} \notin \Gamma, \end{cases} \quad k = 1, \dots, \ell\,. \tag{2.9b}$$

For $\overline{\chi}$ we choose either the mean value of the boundary function χ_0^h or the solution of the proper PDE on the coarsest grid with Dirichlet boundary conditions χ_0^h.

Now, the extension u^h is simply

$$\mathcal{E}_{\overline{\Omega}\Gamma} {'^h} := u^h = \sum_{k=0}^{\ell} u_k^h\,. \tag{2.10}$$

In Algorithm 1 the discrete representation B_{IC}^T of the transposed operator $\mathcal{E}_{\overline{\Omega}\Gamma}^T$ is needed so we prefer the recursive version of definition (2.10):

$$v_0 := u_0^h \tag{2.11a}$$

$$v_k := v_{k-1} + u_k^h \qquad k = 1, \dots, \ell\,, \tag{2.11b}$$

and set $\mathcal{E}_{\overline{\Omega}\Gamma} {'^h} := v_\ell$. Note that $v_k \in \mathbb{W}_k$ is the extension of $\mathcal{Q}_k {'^h}$ on level $k = 0, \dots, \ell$.

3 Matrix Representation of Multilevel Extension Operators

Using the FE isomorphism we change from the operator description of the extension $\mathcal{E}_{\overline{\Omega}\Gamma}$ to the matrix representation. The bilinear FE basis $\Psi^{(k)}$ will be defined similarly to (1.2) and the matrices $I_{C,k}$ denote the proper identities on level k.

The multilevel extension of a function $’^h = \sum_i^{N_{C,\ell}} ’_i \psi_i^{(\ell)} \in \mathbb{V}_\ell$ represented by the vector $\underline{’} \in \mathbb{R}^{N_{C,\ell}}$ into a function $u^h = \sum_i^{N_\ell} u_i \psi_i^{(\ell)} \in \mathbb{W}_\ell$ represented by the vector $\underline{u} \in \mathbb{R}^{N_\ell}$ consists of three steps:

1. Determine the rectangular $N_{C,k} \times N_{C,\ell}$ <u>projection matrix</u> Q_k and define the coefficients of the projection $\mathcal{Q}_k ’^h$ in the FE nodal basis of level k

$$\underline{\beta}_k := Q_k \underline{’} \qquad k = 0, \dots, \ell \,. \tag{3.12}$$

2. According to (2.8) <u>split the vectors</u> $\underline{\beta}_k$ into the coefficients of the multilevel nodal basis presentation of $’^h = \sum_{k=0}^{\ell} \sum_i^{N_{C,k}} \alpha_i^{(k)} \psi_i^{(k)}$. Denoting by $P_{C,k}^{k+1}$, $P_{I,k}^{k+1}$, $P_{IC,k}^{k+1}$ the usual linear FE interpolation matrices on the proper subsets of nodes we can determine the coefficient vectors $\underline{\alpha}_k$

$$\underline{\alpha}_0 \;:=\; \underline{\beta}_0 \tag{3.13a}$$

$$\underline{\alpha}_k \;:=\; \begin{pmatrix} -P_{C,k-1}^k & I_{C,k} \end{pmatrix} \begin{pmatrix} \underline{\beta}_{k-1} \\ \underline{\beta}_k \end{pmatrix} \qquad k = 1, \dots, \ell. \tag{3.13b}$$

3. The coefficients $\underline{v}_k$ of the <u>extensions</u> $v_k = \sum_{i=1}^{N_k} v_i^{(k)} \psi_i^{(k)}$ are determined by

$$\underline{v}_0 = \begin{pmatrix} \underline{v}_{C,0} \\ \underline{v}_{I,0} \end{pmatrix} \;:=\; \begin{pmatrix} I_{C,0} \\ B_{IC,0} \end{pmatrix} \underline{\alpha}_0 \tag{3.14a}$$

$$\underline{v}_k = \begin{pmatrix} \underline{v}_{C,k} \\ \underline{v}_{I,k} \end{pmatrix} \;:=\; \begin{pmatrix} I_{C,k} & P_{C,k-1}^k & 0 \\ 0 & P_{IC,k-1}^k & P_{I,k-1}^k \end{pmatrix} \begin{pmatrix} \underline{\alpha}_k \\ \underline{v}_{C,k-1} \\ \underline{v}_{I,k-1} \end{pmatrix} \tag{3.14b}$$

Set $E_{IC} \underline{’} := \underline{v}_\ell$.
The matrix $B_{IC,0}$ can be chosen as $\frac{1}{N_{C,0}} \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & & \vdots \\ 1 & \cdots & 1 \end{pmatrix}_{N_{I,0} \times N_{C,0}}$, mapping the mean value of the boundary data into the interior. Another approach is the discrete harmonic extension on the coarsest grid with respect to the PDE, i.e., $B_{IC,0} = -K_{I,0}^{-1} K_{IC,0}$.

4 Various Multilevel Extension Techniques

Although having in mind the operator representation from Section 2 we give the following extension techniques mostly in the matrix representation from Section 3. Note, that the theoretical results for the first two techniques below require material coefficients $\lambda(x) = \lambda_i > 0 \quad \forall x \in \overline{\Omega}_i \quad (i = 1, \dots, p)$ whereas the third technique is also available for more general coefficients.

The Hierarchical Extension Technique

Using just the injection from level ℓ to level k for defining the projections Q_k we get exactly the hierarchical extension technique $\widehat{E}_{IC}$ which was first proposed in [HLMN94]. The proofs of the following statements can be found therein.

Setting $B_{IC} := \widehat{E}_{IC}$ the constant c_E in (1.6) behaves like

$$\text{in 2D}: \quad c_E(\widehat{E}_{IC}) = \mathcal{O}(\ln h^{-1}) = \mathcal{O}(\ell) \qquad \text{in 3D}: \quad c_E(\widehat{E}_{IC}) = \mathcal{O}(h^{-1}) . \tag{4.15}$$

This hierarchical extension can also be used as initial guess in an iteration approximating the extension, i.e., $B_{IC} := M_I^s \widehat{E}_{IC} + (I_I - M_I^s) K_I^{-1}(-K_{IC})$ with some iteration operator M_I. If there exists an h-independent positive constant such that $\| M_I \|_{K_I} \leq \eta < 1$ holds, e.g., using multigrid to define M_I, then in the 2D case $s = \mathcal{O}(\ln(\ln h^{-1}))$ iterations are sufficient to achieve an h-independent constant c_E.

The BPX-like Extension Technique

When approaching a BPX splitting with the L_2 ortho-projections of the boundary data $'^h$ the projection matrix will be expressed by $Q_k = M_{k,k}^{-1} M_{k,\ell}$, where the entries in the mass matrix $M_{k,p}$ are defined via the L_2 inner product $m_{i,j}^{(k,p)} = (\psi_i^{(k)}, \psi_j^{(p)})_{L_2}$. Using bilinear FE functions in the 2D case the matrix $M_{k,k}$ possesses exactly 3 non-zero entries per row. Whereas this matrix is rather easy to invert in the 3D case this will require too much arithmetic work.

Therefore a mass lumping or the proposal in [Nep95] for defining the projection operator $\mathcal{Q}_k$ is used. This results in an easily invertible diagonal matrix $\overline{M}_{k,k}$ defining $Q_k := \overline{M}_{k,k}^{-1} M_{k,\ell}$ and leads to the BPX-like extension technique $\overline{E}_{IC}$ for which

$$c_E(\overline{E}_{IC}) = \mathcal{O}(1) \tag{4.16}$$

was proved in [Nep95] for the 2D as well as for the 3D case.

Multilevel Extension Techniques Plus Smoothing

The recursive definition of the extension in (3.14) leads to the idea of an additional improvement of the extensions given above via some linear smoothing procedure $S_{I,k} : \mathbb{R}^{N_{I,k}} \to \mathbb{R}^{N_{I,k}}$, $k = 1, \ldots, \ell$ with the properties

$$\begin{array}{lllll} \text{High frequencies}: & \| S_{I,k} \underline{v}^h \|_{K_I} & \leq & \varrho_k \| \underline{v}^h \|_{K_I} & \forall \Phi_I \underline{v}^h \in \mathbb{W}_k \setminus \mathbb{W}_{k-1} \\ \text{Low frequencies}: & \| S_{I,k} \underline{v}^h \|_{K_I} & \leq & \sigma_k \| \underline{v}^h \|_{K_I} & \forall \Phi_I \underline{v}^h \in \mathbb{W}_k , \end{array} \tag{4.17}$$

where the number of smoothing sweeps is denoted by ν_k. We require smoothing factors $\varrho_k \leq \sigma_k \leq 1$ independent of $h = 2^{-l}$. So, definition (3.14b) changes into

$$\begin{pmatrix} \underline{v}_{C,k} \\ \underline{v}_{I,k} \end{pmatrix} := \begin{pmatrix} I_{C,k} & 0 \\ -(I_{I,k} - S_{I,k}^{\nu_k}) K_{I,k}^{-1} K_{IC,k} & S_{I,k}^{\nu_k} \end{pmatrix} \begin{pmatrix} I_{C,k} & P_{C,k-1}^k & 0 \\ 0 & P_{IC,k-1}^k & P_{I,k-1}^k \end{pmatrix} \begin{pmatrix} \underline{\alpha}_k \\ \underline{v}_{C,k-1} \\ \underline{v}_{I,k-1} \end{pmatrix} \tag{4.18}$$

Using the hierarchical extension together with the smoothing, i.e., setting $B_{IC} = \widehat{E}_{IC}(\nu)$, it was shown in [Haa97] that in the 2D case

$$c_E(\widehat{E}_{IC}(\nu)) \leq c \left(1 + \sqrt{\ell} \cdot \sqrt{\sum_{k=1}^{\ell} \varrho_k^{2\nu_k} \prod_{j=k+1}^{\ell} \sigma_j^{2\nu_j}} \right) = \mathcal{O}(\ell) \tag{4.19}$$

holds with a positive and h-independent constant c. In comparison to the hierarchical extension the order remains the same but the constants hidden in that order statement are partially controlled by the smoothing. Now, $\nu_k = \mathcal{O}(\ln(\ln h^{-1}))$ iterations of the smoothing procedure $S_{I,k}$ are sufficient to achieve an h-independent constant c_E so that an additional iteration, as in the hierarchical extension technique, is no longer needed.

In the BPX-like extension additional smoothing sweeps are also feasible but the theory for this is still open. In that case the parameter σ_k will have more influence on the behavior of the constant c_E.

5 An Algorithmic Improvement of the Preconditioner

The improvement of Algorithm 1 is based upon three observations:

A) In Algorithm 1 the matrices B_{IC}^T and C_I^{-1} are applied to the same vector $\underline{r}_I$.

B) In the transposed operation to (3.14b) used in B_{IC}^T,

$$\begin{pmatrix} \underline{\alpha}_k \\ \underline{v}_{C,k-1} \\ \underline{v}_{I,k-1} \end{pmatrix} := \begin{pmatrix} I_{C,k} & 0 \\ \left(P_{C,k-1}^k\right)^T & \left(P_{IC,k-1}^k\right)^T \\ 0 & \left(P_{I,k-1}^k\right)^T \end{pmatrix} \begin{pmatrix} \underline{v}_{C,k} \\ \underline{v}_{I,k} \end{pmatrix} ,$$

the last row is simply the usual linear restriction from the finer to the coarser grid. A similar observation for the extension technique plus smoothing in (4.18) leads to

$$\underline{v}_{I,k-1} := \left(P_{I,k-1}^k\right)^T \left(S_{I,k}^T\right)^{\nu_k} \underline{v}_{I,k} \; . \tag{5.20}$$

C) Perform in a multigrid algorithm at level k ν_k smoothing sweeps with the iteration matrix $\overset{*}{S}_{I,k}$, ($\overset{*}{S}_{I,k}$ means the adjoint matrix to $S_{I,k}$ in the $K_{I,k}$-energy inner product) together with calculation and restriction of the defect. Then that brief algorithm,

$$\underline{w}_{I,k} := (I_{I,k} - (\overset{*}{S}_{I,k})^{\nu_k})K_{I,k}^{-1}\underline{f}_{I,k} \quad \text{and} \quad \underline{d}_{I,k-1} := \left(P_{I,k-1}^k\right)^T (\underline{f}_{I,k} - K_{I,k}\underline{w}_{I,k}) \; ,$$

can be simplified into

$$\underline{d}_{I,k-1} := \left(P_{I,k-1}^k\right)^T \left(S_{I,k}^T\right)^{\nu_k} \underline{f}_{I,k} \; . \tag{5.21}$$

Now, choosing $\overset{*}{S}_{I,k}$ as pre-smoother and $S_{I,k}$ as post-smoother in a multigrid algorithm defining C_I we preserve the required symmetry and may use restriction and pre-smoothing from the implementation of the transposed extension B_{IC}^T. When, additionally, in that extension a coarse grid solver is used more than half of the algorithmic work can be saved when applying C_I^{-1}. This algorithmic improvement does not depend on the choice of the projections Q_k !

Numerical tests using this improved algorithm together with the hierarchical extension technique can be found in [Haa97].

6 A test example

For checking the theoretical results concerning the behavior of the constant c_E it is necessary that the spectral equivalence constants in (1.5) are independent of h. Therefore, the test example consists of the PDE

$$-\Delta\, u(x) \;=\; 1 \quad \text{in}\; \Omega = (0,1) \times (0,0.5) \qquad \text{and} \qquad u(x) = 0 \quad \text{on}\; \partial\Omega \;\;,$$

where the domain $\overline{\Omega}$ was decomposed into two squares. Using in the preconditioner C (1.4) Dryja's approach [Dry82] as Schur complement preconditioner C_C and exact solvers for $C_{I,i}^{-1}$, we achieve h-independent spectral equivalence constants in (1.5), so the condition number of the preconditioned system $\kappa(C^{-1}K)$ is influenced only by the extension B_{IC}.

The discrete system (1.3) was solved with a preconditioned parallelized CG until a relative accuracy of 10^{-6} measured in the $\|\cdot\|_{KC^{-1}K}$-norm of the error was reached. Due to estimates (1.7) the number of CG iterations behaves like $\mathcal{O}(c_E^2)$.

On level 0 with the discretization parameter $h = 0.25$, i.e., just one node on the interface between the two subdomains, the automatic mesh generator produces a triangular mesh with 4 inner nodes per subdomain. All finer meshes were produced by simply subdividing each triangle into 4 congruent ones. The Gauß-Seidel smoother $S_{I,k}$ applied $\nu = \nu_k$-times $(k = 1, \ldots, \ell)$ and a coarse grid solver were used in the extension B_{IC}. Whereas in Table 1 the iteration numbers connected with the hierarchical splitting grow linearly with the number of levels ℓ the iteration numbers of the BPX-like splitting tend to an upper bound. This confirms the theoretical results from (4.15), (4.16) and (4.19). Obviously the additional smoothing sweeps decrease the number of iterations and also the condition number $\kappa(C^{-1}K)$ rapidly.

Table 1 Number of CG iterations for the test example using 2 processors

Splitting	ν	$\ell=0$	$\ell=1$	$\ell=2$	$\ell=3$	$\ell=4$	$\ell=5$	$\ell=6$
hier.	0	2	7	11	17	24	30	36
hier.	1	2	6	8	11	13	16	19
hier.	2	2	5	7	8	10	13	14
BPX-like	0	2	7	8	11	12	13	13

7 Final Remarks

Although the development was done on rather simple operators, the results carry over to systems of symmetric elliptic second order PDE s resulting in symmetric $\mathbb{V}_0$-elliptic and $\mathbb{V}_0$-bounded bilinear forms. The extension techniques used in the test example work also very well in more challenging examples and result in a faster solver when the algorithmic improvement from Paragraph 4 is used; see also [Haa97].

In the latter paper, the full proof for estimate (4.19) can be found and it was shown that this estimate holds for the whole range from extension without smoothing to the exact harmonic extension. Especially smoothing controls similar to the multigrid slash cycle and generalized slash cycle were investigated, so that $O(\ln \ell)$ smoothing sweeps are sufficient to achieve a condition number $\kappa(C^{-1}K) = O(1)$. The algorithmic improvement from Paragraph 5 was derived in a more heuristic way. The electrical machine with jumping coefficients and a complicated geometry proved that the new extension method is also successfully applicable to practical problems.

The implementation and theoretical analysis of the BPX-like extension together with smoothing sweeps will be done in a forthcoming paper. Also, a comparison to other extension techniques proposed in [Nep91, Che93, BPV96] should be done on a more challenging example, with respect to the CPU time needed to solve (1.1).

REFERENCES

[Boe89] Boergers M. (1989) The Neumann–Dirichlet domain decomposition method with inexact solvers on the subdomains. *Numerische Mathematik* 55(2): 123–136.

[BPS89] Bramble J., Pasciak J., and Schatz A. (1986, 1987, 1988, 1989) The construction of preconditioners for elliptic problems by substructuring I – IV. *Mathematics of Computation* 47, 103–134, 49, 1–16, 51, 415–430, 53, 1–24.

[BPV96] Bramble J., Pasciak J., and Vassilev A. (1996) Analysis of non-overlapping domain decomposition algorithms with inexact solves. Technical report. available via http://www.math.tamu.edu/~james.bramble/papers.html.

[BPX90] Bramble J., Pasciak J., and Xu J. (1990) Parallel multilevel preconditioners. *Mathematics of Computation* 55(191): 1–22.

[Che93] Cheng H. (1993) *Iterative Solution of Elliptic Finite Element Problems on Partially Refined Meshes and the Effect of Using Inexact Solvers.* PhD thesis, Courant Institute of Mathematical Science, New York University.

[Dry82] Dryja M. (1982) A capacitance matrix method for Dirichlet problems on polygonal regions. *Numerische Mathematik* 39(1): 51–64.

[Haa97] Haase G. (May 1997) Hierarchical extension operators plus smoothing in domain decomposition preconditioners. *Applied Numerical Mathematics* 23(3).

[HL92] Haase G. and Langer U. (1992) The non-overlapping domain decomposition multiplicative Schwarz method. *International Journal of Computer Mathemathics* 44: 223–242.

[HLM91] Haase G., Langer U., and Meyer A. (1991) The approximate Dirichlet domain decomposition method. Part I: An algebraic approach. Part II: Applications to 2nd-order elliptic boundary value problems. *Computing* 47: 137–151 (Part I), 153–167 (Part II).

[HLMN94] Haase G., Langer U., Meyer A., and Nepomnyaschikh S. (1994) Hierarchical extension operators and local multigrid methods in domain decomposition preconditioners. *East-West Journal of Numerical Mathematics* 2: 173–193.

[Nep91] Nepomnyaschikh S. (1991) Method of splitting into subspaces for solving elliptic boundary value problems in complex-form domains. *Sov. J. Numer. Anal. Math. Modelling* 6(2).

[Nep95] Nepomnyaschikh S. (1995) Optimal multilevel extension operators. Report 95-3, TU Chemnitz.

[Osw94] Oswald P. (1994) *Multilevel Finite Element Approximation.* Teubner, Stuttgart.

[SBG96] Smith B., Bjorstad P., and Gropp W. (1996) *Domain Decomposition : parallel methods for elliptic partial differential equations.* Cambridge University Press.

[TCK92] Tong C., Chan T., and Kuo C. J. (1992) Multilevel filtering preconditioners: Extensions to more general elliptic problems. *SIAM J. Sci. Stat. Comput.* 13: 227–242.

[Xu89] Xu J. (1989) Theory of multilevel methods. Technical Report AM48, Department of Mathematics, Penn State University.

[Yse86] Yserentant H. (1986) On the multi-level splitting of finite element spaces. *Numer. Math.* 49(4): 379–412.

43

Defect correction for Boussinesq Flow

Wilhelm Heinrichs

1 Introduction

A defect correction method for the convection-diffusion equation is presented. In the domain decomposition context the presented technique can be used as a preconditioner on each subdomain. The discretization is performed by second-order finite difference schemes (β-schemes) where the second-order upstream scheme is combined with the standard central scheme. Higher order discretizations with spectral methods are also considered. For preconditioning the usual first-order upstream scheme is employed. The defect correction iteration is used for relaxation inside a multigrid procedure. It is shown that the smoothing analysis yields rather pessimistic results. In the practical computation the discretization error is reached in two V-cycles. Numerical results are presented which demonstrate the high efficiency of our treatment. The convection-diffusion equation yields a good model for the numerical solution of the Navier-Stokes equations with high Reynolds numbers. In many applications the second-order accuracy is necessary in order to get a realistic impression of the flow.

The β-schemes were previously analyzed by Desideri & Hemker [DH92] and Luh [Luh92]. For $\beta = 1$ we obtain the standard second-order upstream scheme and for $\beta = 0$ the central scheme. $\beta = \frac{1}{2}$ results in *Fromm's* scheme. For $\beta = \frac{1}{3}$ we obtain the *upwind biased* scheme which is of third-order accuracy. Since an iterative solver for these higher order schemes yields bad convergence factors we propose a defect correction procedure. We have had good experience with this method for spectral discretizations. Here the higher order scheme is also preconditioned by the standard first-order upwind scheme. We investigate the smoothing properties of this procedure for the convection equation. This method is used for relaxation in a multigrid cycle. In the spectral scheme the solution is approximated by Chebyshev polynomials. By a Fourier analysis it can be shown that the eigenvalues of the preconditioned operator are bounded but complex. Hence one has to employ a nonsymmetric matrix iteration for the solution. We recommend the GMRES iteration which belongs to the residual minimization methods. Clearly, for the general convection-diffusion problem the first derivatives have to be approximated according to the sign of the coefficients. Therefore

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

for the iterative solution we recommend flow-directed schemes. Since the Chebyshev nodes are dense near the boundary it is necessary to use line Gauss-Seidel relaxation (in an alternating manner). Finally this iterative solver is applied to the Boussinesq flow problem in vorticity-streamfunction formulation with high Rayleigh numbers.

2 Convection-diffusion Problem

Here we consider convection-diffusion problems which can in scalar, constant-coefficient form be written as

$$\begin{aligned} -\epsilon\Delta u + au_x + bu_y &= f \text{ in } \Omega = (-1,1)^2, && (2.1)\\ u &= g \text{ on } \partial\Omega, && (2.2)\end{aligned}$$

where $\epsilon = \frac{1}{Re}$ and Re denotes the Reynolds number. f is defined in Ω, and g is defined on $\partial\Omega$. a and b denote given constants. Such problems arise after a linearization of the Navier-Stokes equations or Boussinesq flow problems (see Section 3). The part $-\epsilon\Delta u$ denotes the diffusive part and $au_x + bu_y$ denotes the convective part of the above equation. Here we are mainly interested in convection-dominated flows where $\epsilon \ll h$ or $\epsilon \ll N^{-2}$. Here h denotes the step size of the finite difference (FD) scheme and N the maximal degree of the polynomials in a spectral scheme. It is well known that discretizations for this type of problem are in general unstable. One possibility to avoid the phenomenon of instability is to use upstream discretization for u'. An obvious disadvantage of this scheme lies in the fact that the method now becomes only first-order accurate. Hence it makes sense to use the first-order upstream scheme only as a preconditioner for a higher order scheme. We analyze the preconditioning properties of this method for the following higher order schemes:

- β-schemes, $\beta \in [0,1]$,
- spectral methods.

The first-order upstream scheme is explicitly given by the upstream operator L_h^1 (here in 1D):

$$L_h^1 \;\cong\; \frac{1}{h}[-1\ \underline{1}\ 0]\,.$$

The second-order hybrid scheme is the β-scheme L_h^β, which is a combination of the standard second-order upstream scheme $L_h^{su,2}$ and the second-order central scheme $L_h^{ce,2}$:

$$L_h^\beta \;=\; \beta L_h^{su,2} \;+\; (1-\beta)L_h^{ce,2}, \quad \beta \in [0,1]. \qquad (2.3)$$

For all β we obtain at least second-order discretizations. For $\beta = 0$ the method becomes unstable. Especially, for $\beta = \frac{1}{3}$ we obtain a third-order scheme. Here we study the preconditioning properties of L_h^1 for the 2D convection operator

$$au_x \;+\; bu_y.$$

The defect correction iteration is defined by the operator

$$M_h \;=\; I_h \;-\; \omega\left(L_h^1\right)^{-1} L_h^\beta, \quad \beta \in [0,1].$$

Here ω denotes a relaxation parameter which should accelerate the convergence speed. By a Fourier analysis for the Fourier components $\underline{\theta}$ ($\underline{\theta} = \theta_1$ in $1D$, $\underline{\theta} = (\theta_1, \theta_2)$ in $2D$) the above operator M_h leads to the *amplification factor*

$$\mu(\underline{\theta}) \;=\; 1 - \omega \frac{\lambda^\beta(\underline{\theta})}{\lambda^1(\underline{\theta})},$$

where $\lambda^\beta(\underline{\theta})$ and $\lambda^1(\underline{\theta})$ denote the factors of the Fourier analysis for the operators L_h^β and L_h^1. Now the convergence factor ρ of the defect correction procedure is defined as the supremum of $\mu(\underline{\theta})$ taken over all frequencies $\underline{\theta} \in (-\pi, \pi]$ in 1D and $\underline{\theta} \in (-\pi, \pi]^2$ in 2D:

$$\rho(M_h) \;:=\; \sup_{\underline{\theta} \in (-\pi,\pi]^2} |\mu(\underline{\theta})|.$$

Furthermore, we are interested in defect correction as a smoother in a multigrid procedure. The efficiency of a smoother can be measured by means of the *smoothing rate* μ_β. This rate can be obtained by taking the above supremum only for the high frequencies $|\underline{\theta}| := \max(|\theta_1|, |\theta_2|) \in (\frac{\pi}{2}, \pi]$:

$$\mu_\beta \;:=\; \sup_{\underline{\theta} \in (\frac{\pi}{2},\pi]} |\mu(\underline{\theta})|.$$

The Fourier analysis of Luh [Luh92] makes it clear that for all $\beta \in [0, 1]$, ω and independent of the *alignment* $\frac{b}{a}$ the prediction

$$\rho(M_h) \;=\; 1$$

holds.

However, in the multigrid procedure we are more interested in the damping only of the high frequencies. Here we consider the special cases $\beta = 1$ and $\beta = \frac{1}{2}$. From the analysis in [Luh92] we observe:

- $\beta = 1$, $\omega_{opt} = 0.68$: $\mu_1 \leq 0.77$,
- $\beta = \frac{1}{2}$, $\omega_{opt} = 1.00$: $\mu_{\frac{1}{2}} \leq 0.72$.

For $\beta = 1$ the optimal parameter ω is about 0.68 for all alignments $\frac{b}{a}$. The corresponding smoothing rate μ_1 is decreasing for increasing alignment. The maximum is attained at $\frac{b}{a} = 0.1$ where $\mu_1 = 0.77$. For $\beta = \frac{1}{2}$ the value $\omega_{opt} = 1$ yields a quite good choice for all alignments $\frac{b}{a}$. For $\beta = 0$ the smoothing rate μ_0 is always equal to 1 (independent of the parameter choice of ω). From these considerations it becomes clear that the smoothing rates are rather bad compared to the usual rates for symmetric problems (e.g., the Poisson problem).

Here we apply a standard multigrid method. The transfer operators are given by *full weighting* restriction and *bilinear* interpolation. For relaxation we choose the already mentioned *Richardson* iteration with *defect correction.* It is explicitly defined as follows:

$$u_h^{j+1} \;=\; u_h^j \;-\; \omega \left(L_h^1\right)^{-1} \left(L_h^\beta u_h^j - f_h\right)$$

for $j = 0, 1, 2, \ldots$. u_h^0 denotes an initial approximation, which is chosen to be identically zero. ω denotes the relaxation parameter. Optimal choices are given by the smoothing analysis. Instead of solving the first-order problem relative to L_h^1 exactly we employ a *lexicographic* Gauß-Seidel step, which nearly yields an exact solver if $a,\ b > 0$. If a and b have different sign then after a renumbering of the grid points the same effect can be achieved. In case of variable coefficients a and b which change sign one has to use the *flow directed point relaxation.* Here we consider a multigrid method with 7 grids and corresponding step sizes $h_\nu = \frac{1}{2^\nu}$, $\nu = 1, \ldots, 7$. In general, we employ a *V–cycle.* Other cycle structures as *W–cycle, F–cycle* or the *full multigrid* technique could not improve the convergence speed significantly. We employ two relaxations before and one after the coarse grid correction. The results are compared with the pure Richardson iteration without using multigrid. The absolute errors between the exact solution and the iterates are measured in the discrete L^1 and L^2 norms: $L1,\ L2$. By $Q(L1),\ Q(L2)$ we denote the quotient of the errors for two successive iterates. Numerical results are provided for the example where the exact solution is given by

$$u(x,y) = \sin\left(8\pi(y - \frac{b}{a}x)\right),$$

where $a = 4,\ b = 1$ and $\epsilon = 10^{-6}$ (see Y. Luh [Luh92]). The right hand side f and the Dirichlet boundary conditions are determined by u. We present results for $\beta = 1$. The corresponding numerical results are given in Table 1.

Table 1. $\beta = 1$, V–cycle

IT	$L1$	$L2$	$Q(L1)$	$Q(L2)$
1	$4.28 \cdot 10^{-2}$	$5.42 \cdot 10^{-2}$	0.067	0.076
2	$2.44 \cdot 10^{-2}$	$3.04 \cdot 10^{-2}$	0.569	0.560
3	$2.42 \cdot 10^{-2}$	$3.01 \cdot 10^{-2}$	0.994	0.989

The numerical results show the highly improved efficiency of the V-cycle. There is nearly no improvement by using other cycle structures. In particular, the first rate of the multigrid scheme is very small and yields already an error which is nearly equal to the discretization error. Hence in general it can be seen that 2 or 3 V-cycles are enough to reach the truncation error. So the smoothing analysis gives rather pessimistic results. In practice, the discretization error is reached very fast. By comparing the results for different step sizes we also confirm at least second-order accuracy. For $\beta = \frac{1}{3}$ we obtain a third-order method. Here we obtain the most precise results. For increasing β the results become less accurate.

Let us now consider the case of variable coefficients $a,\ b$, i.e., $a = a(x,y),\ b = b(x,y)$. For the iterative solution we recommend *flow directed schemes.* For smoothing it is recommended to use alternate iterations of FDHI (Flow Directed Horizontal Iterations) and FDVI (Flow Directed Vertical Iterations). In the literature this combination is called FDHVI (see [DM92], [HIK88]). The iterative scheme FDHI is a variant of line Gauss-Seidel relaxation. Han et al. [HIK88] describe a procedure based on directed graphs to partition and order the unknowns of the Gauss-Seidel process. This is performed by inspection of the coefficient matrix. Nevertheless, this algorithm is expensive for nonlinear problems, like those coming from the Navier-Stokes or Boussinesq equations, when the coefficients are solution-dependent and require the reconstruction of the directed graph several times. The penalty for such a choice is proportional to the number of mesh points. Here the FDHVI scheme is applied to the Boussinesq flow problem.

3 The Boussinesq Flow Problem

The problem specifically considered here is that of the two-dimensional flow of a Boussinesq fluid of Prandtl number $Pr = 0.71$ (i.e., air) in an upright square cavity (see [BWD90], [Dav83]). The walls are non-slip and impermeable. The horizontal walls are adiabatic and the vertical sides are at fixed temperatures. In addition to the Navier-Stokes equations we have one further equation for the temperature T. By Ra we denote the Rayleigh number. The Boussinesq flow problem in vorticity-streamfunction formulation reads as follows:

$$\begin{aligned} 4\Delta\psi + \omega &= 0 \text{ in } \Omega = (-1,1)^2, && (3.4)\\ -2Pr\Delta\omega + \frac{\partial}{\partial x}(v_1\omega) + \frac{\partial}{\partial y}(v_2\omega) &= RaPr\frac{\partial T}{\partial x} \text{ in } \Omega, && (3.5)\\ -2\Delta T + \frac{\partial}{\partial x}(v_1 T) + \frac{\partial}{\partial y}(v_2 T) &= 0 \text{ in } \Omega. && (3.6) \end{aligned}$$

As usual $(v_1, v_2)^t$ denotes the velocity. ψ fulfills homogeneous Dirichlet boundary conditions and T fulfills mixed Dirichlet/Neumann boundary conditions. The homogeneous Neumann boundary conditions correspond to the fact that the horizontal walls are adiabatic.

Now the equations (4)–(6) are linearized by a Quasi-Newton method, where the velocity from the previous iteration is employed. The linearized system is then approximately solved by a spectral multigrid (SMG) method (see [Hei88a], [Hei88b], [Hei92], [Hei93]). In the spectral scheme the solution is approximated by polynomials in $\mathbf{P}_N$, $N \in \mathbf{N}$ where $\mathbf{P}_N$ denotes the space of polynomials of degree $\leq N$. The discretization is performed by a pseudo spectral (or collocation) method in the Chebyshev-Gauss-Lobatto nodes. In the SMG method we use the same components as already introduced. We employ the FDHVI iteration for preconditioning. In order to handle the complex eigenvalues of the preconditioned spectral operator we employ nonsymmetric matrix iterations. Here we choose the GMRES iteration.

By using these components we numerically calculated for various Rayleigh numbers and mesh sizes the following quantities:

$|\psi|_{mid}$: absolute value of the streamfunction at the midpoint of the cavity,
$|\psi|_{max}$: maximum absolute value of the streamfunction,
$v_{1,max}$: maximum horizontal velocity on the vertical mid-plane of the cavity,
$v_{2,max}$: maximum horizontal velocity on the horizontal mid-plane of the cavity.

The local heat flux in a horizontal direction at any point in the cavity is given by

$$Q \;:=\; v_1 T \;-\; 2\frac{\partial T}{\partial x}.$$

Let us further introduce the following Nusselt numbers:

$\overline{Nu} \;:=\; \frac{1}{4}\int_{-1}^{1}\int_{-1}^{1} Q(x,y)dxdy$: average Nusselt number throughout the cavity,
$Nu_{\frac{1}{2}} \;:=\; \frac{1}{2}\int_{-1}^{1} Q(0,y)dy$: average Nusselt number on the vertical mid-plane,
$Nu_0 \;:=\; \frac{1}{2}\int_{-1}^{1} Q(-1,y)dy$: average Nusselt number on the vertical boundary.

The above integrals in the definition of $\overline{Nu}$, $Nu_{\frac{1}{2}}$ and Nu_0 are evaluated by the Clenshaw-Curtis quadrature. In Table 2, we present the numerical results for the Rayleigh numbers $Ra = 10^5$. The numerical results are in good accordance with the results obtained in [Dav83].

Table 2. Results for $Ra = 10^5$.

N	$\|\psi\|_{mid}$	$\|\psi\|_{max}$	$v_{1,max}$	$v_{2,max}$	$\overline{Nu}$	$Nu_{\frac{1}{2}}$	Nu_0
8	14.3409	18.8519	37.8844	40.2643	4.4140	4.7345	4.7590
16	11.3720	12.3330	36.3420	61.3420	4.5030	4.5061	4.5313
24	9.1600	9.6530	34.6320	67.9120	4.5100	4.5120	4.5231

REFERENCES

[BWD90] Behnia M., Wolfstein M., and Davis G. D. V. (1990) A stable fast marching scheme for computational fluid mechanics. *Int. J. for Num. Meth. in Fluids* 10: 607+.

[Dav83] Davis G. D. V. (1983) Natural convection of air in a square cavity: a benchmark numerical solution. *Int. J. for Numer. Meth. in Fluids* 3: 249+.

[DH92] Desideri J. and Hemker P. (1992) Analysis of the convergence of iterative implicit and defect-correction algorithms for hyperbolic systems. Research Report 9004, Centre for Mathematics and Computer Science.

[DM92] Deville M. and Mund E. (1992) Finite element preconditioning of collocation schemes for advection-diffusion equations. In Beauwens R. and de Groen P. (eds) *Proceedings of the IMACS International Symposium on Iterative Methods in the Linear Algebra*, pages 203–208. Elsevier Science Publishing Company, Brussels.

[Hei88a] Heinrichs W. (1988) Line relaxation for spectral multigrid methods. *J. Comp. Phys.* 77: 166+.

[Hei88b] Heinrichs W. (1988) Multigrid methods for combined finite difference and fourier problems. *J. Comp. Phys.* 78: 424+.

[Hei92] Heinrichs W. (1992) Spectral multigrid techniques for the stokes problem in streamfunction formulation. *J. Comp. Phys.* 102: 310+.

[Hei93] Heinrichs W. (1993) Spectral multigrid techniques for the navier-stokes equations. *Comput. Meth. Appl. Mech. Eng.* 106: 297+.

[HIK88] Han H., Il'in V., and Kellogg R. (1988) Flow directed iterations for advection dominated flow. In Ben-yu G., Miller J., and Shi Z.-c. (eds) *BAIL V, Proceedings of the Fifth Int. Conf. on Boundary and Interior Layers - Computational and Asymptotic Methods.* Shanghai, China.

[Luh92] Luh Y. (1992) *Diskretisierungen und Mehrgitteralgorithmen zur Lösung hyperbolischer Differentialgleichungen, am Beispiel der Wellengleichung, der Advektionsgleichung und der verallgemeinerten Stokes-Gleichungen.* GMD-Bericht Nr. 205. Oldenbourg-Verlag, Oldenbourg.

44

Adaptive Meshes for the Spectral Element Method

Li-Chieh Hsu and Catherine Mavriplis

1 Introduction

The spectral element method [Pat84] is a high order domain decomposition method for the solution of nonlinear time-dependent partial differential equations. The method has been successfully used in the solution of the Navier-Stokes equations for direct simulation of many complex fluid flows e.g., [Kar90, FR94]. Although the method is, in theory, very powerful for complex phenomena such as transitional flows, practical implementation of domain decomposition for optimal resolution of complex features limits its performance. For instance, it is hard to estimate the appropriate number of elements for a specific case. *A priori* selection of regions to be refined or coarsened is difficult especially as the flow becomes more complex and memory limits of the computer are stressed.

In this paper we present an adaptive spectral element method in which decomposition of the domain is automatically determined in order to capture underresolved regions of the domain and to follow regions requiring high resolution as they develop in time. The objective is to provide the best and most efficient solution to a time-dependent nonlinear problem by continually optimizing resource allocation.

Previously [Mav94], the advantages of such an adaptive scheme were shown to be significant: singularities, thin internal or boundary layers may be resolved by automatic detection and refinement. In [Mav94], the relative merits of the refinement options available for spectral elements were demonstrated in one dimension and a simple two-dimensional example was given. In this paper, we offer a more complete description of the two-dimensional adaptive method. The spectral element method provides two modes of resolution refinement: increase in the number of elements (h-refinement) and increase in the polynomial order of the basis functions on each element (p-refinement). In this paper, for simplicity, the refinement options are limited to h-refinement only.

In the following, the spectral element method is briefly reviewed and the adaptive algorithm is described. Example calculations are then presented for two-dimensional heat conduction and Stokes flow problems.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 Discretization

Our objective is to simulate complex flows by solving the incompressible Navier-Stokes equations:

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \boldsymbol{u},$$
$$\nabla \cdot \boldsymbol{u} = 0,$$

where $\boldsymbol{u}$ is the velocity vector, p is the pressure, ρ is the density and ν is the kinematic viscosity. A time splitting scheme [OK80]:

$$\frac{\hat{\boldsymbol{u}} - \boldsymbol{u}^n}{\Delta t} = \sum_{r=0}^{2} \beta_r (-\boldsymbol{u} \cdot \nabla \boldsymbol{u})^{n-r},$$

$$\nabla^2 p^{n+1} = \frac{\rho}{\Delta t} \nabla \cdot \hat{\boldsymbol{u}},$$

$$\frac{\hat{\hat{\boldsymbol{u}}} - \hat{\boldsymbol{u}}}{\Delta t} = -\frac{1}{\rho}\nabla p^{n+1},$$

$$\frac{\boldsymbol{u}^{n+1} - \hat{\hat{\boldsymbol{u}}}}{\Delta t} = \nu \nabla^2 \boldsymbol{u}^{n+1},$$

where $\hat{\boldsymbol{u}}$ and $\hat{\hat{\boldsymbol{u}}}$ are intermediate time step values of velocity between the nth and $n+1$st time steps, is used to treat the nonlinear terms explicitly, in this case by third order Adams-Bashforth with coefficients β_r, while treating the diffusion and pressure terms implicitly. The pressure and velocities are governed by Helmholtz problems which are discretized by the spectral element method as follows.

For a model Helmholtz equation

$$(\nabla^2 - \lambda^2)\phi = g,$$

we take the variational form

$$-\int \nabla\phi \cdot \nabla\psi dx - \lambda^2 \int \phi\psi dx = \int g\psi dx \;\; \forall \psi$$

and substitute discrete approximations to all variables following

$$\phi_h^k = \sum_{p=0}^{N} \sum_{q=0}^{M} \phi_{pq}^k h_p(r) h_q(s), \tag{2.1}$$

where the h functions are the Lagrangian interpolants based on the orthogonal set of Legendre polynomials of high degree N or M and r, s are the local coordinates on each element k. Performing Gauss-Lobatto quadrature and summing contributions from adjacent elements we obtain the global matrix equation

$$(A - \lambda^2 B)\phi = Bg,$$

where A is the discrete Laplacian operator and B is the mass matrix. The matrix equation is solved by preconditioned conjugate gradient iteration.

For adaptivity and local refinement, the nonconforming formulation is advantageous since it allows elements to abut in arbitrary manners. For example, a conforming mesh is shown in Fig. 2(a) and a nonconforming mesh is shown in Fig. 3(a). In this case, the matrix equation becomes

$$Q^T(A-\lambda^2B)Q\phi = Q^TBg$$

where Q is a transformation matrix which provides an $\mathcal{L}^2$ minimization of the jump in variables on the nonconforming interface [MMP89].

3 The Adaptive Method

The adaptive spectral element method is designed to have low cost and high efficiency in solving complex time-dependent physical problems. The adaptivity is based on error estimators which determine which regions need more resolution. The solution strategy is as follows: compute an initial solution with a suitable initial mesh, estimate errors in the solution locally in each element, modify the mesh according to the error estimators, interpolate the old mesh solutions onto the new elements, and resume the numerical solution process. This solution process is visualized in the flowchart below (Fig. 1).

The error estimators are based on a posteriori estimates of the $\mathcal{L}^2$ and $\mathcal{H}^1$ errors in the spectral representation of the solution on each element [Mav90]. For simplicity, we present the one-dimensional $\mathcal{L}^2$ error estimate on element k, ϵ^k_{est}, as

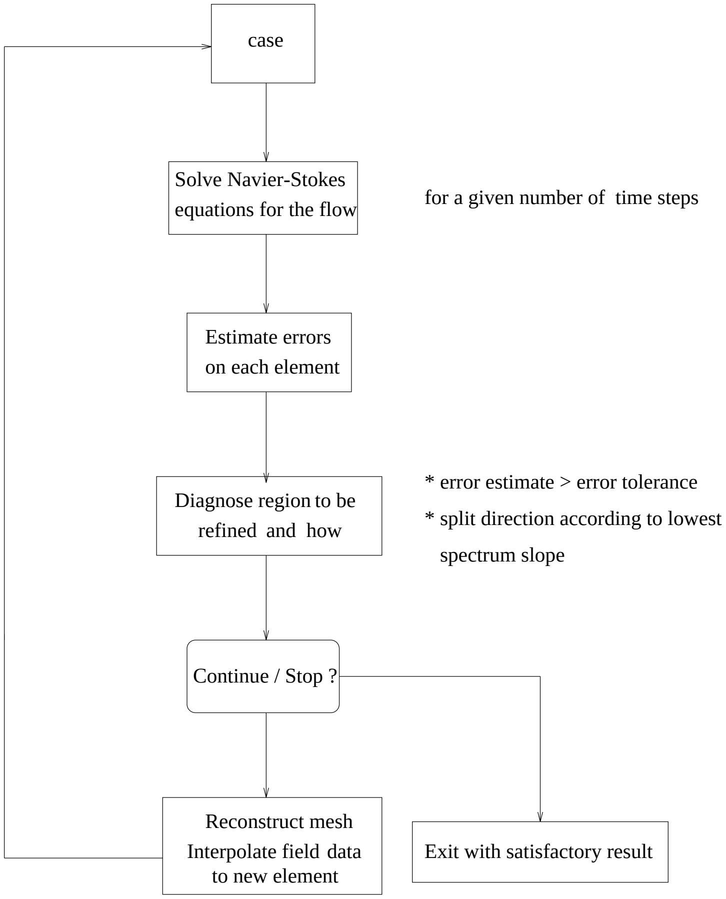
$$\epsilon^k_{est} = \left(\frac{{a^k_N}^2}{\frac{2N+1}{2}} + \int_{N+1}^{\infty} \frac{(a^k(n))^2}{\frac{2n+1}{2}} dn \right)^{\frac{1}{2}} .$$

a^k_n is the one-dimensional spectrum of the numerical solution ϕ^k_h, defined by the elemental spectral discretization (similar to Eqn. 2.1 but in one dimension) rewritten in terms of the Legendre polynomials P_n of order n as

$$\phi^k_h = \sum_{n=0}^{N} a^k_n P_n(r).$$

The $a^k(n)$ function in the error estimate is a model least squares best fit to the last four spectrum points ($n = N-3, N$), which is used to extrapolate the spectrum to infinity in order to estimate the truncation error. In two dimensions, the spectrum is a two-dimensional tensor a^k_{nm}. The two-dimensional elemental error estimate ϵ^k_{est} for element k is made up of the sum of all $\epsilon^k_{m\ est}$ and $\epsilon^k_{n\ est}$ for each $m = 0, M$ and $n = 0, N$, respectively, as well as an extrapolation for $n > N$ **and** $m > M$. In practice, we find that these error estimators are very robust and quite accurate as shown in [Mav90, Mav94] and in the following examples. The value of the estimate ϵ^k_{est} on each element is used to decide whether to adapt or not: if it is greater than a chosen upper error tolerance level, then the particular element k is refined. Similarly, if it is lower than a chosen lower error tolerance level than the element may be coarsened. Coarsening with h-refinement can be difficult to be made robust and, hence, was not implemented in this paper. While local error estimates are used to determine which

Figure 1 Adaptive refinement algorithm


case
Solve Navier-Stokes equations for the flow
for a given number of time steps
Estimate errors on each element
Diagnose region to be refined and how
* error estimate > error tolerance
* split direction according to lowest spectrum slope
Continue / Stop ?
Reconstruct mesh
Interpolate field data to new element
Exit with satisfactory result

regions should be refined, the slopes of the spectrum are used as criteria in choosing the direction of the h-refinement: a lower slope in the r direction indicates that the quality of the solution is poorer in the r direction than in the s direction and hence the element is split in the r direction rather than in the s direction.

Once the new grid has been defined, the current (old) time step solution must be interpolated onto the new topology. The location of the new elements and their Gauss-Lobatto collocation points are determined in the old mesh. Then, through the use of the Lagrangian interpolants from the old mesh, the calculation of the new values of the solution on the Gauss-Lobatto collocation points of the new grid is straightforward. The high order of the spectral element method minimizes errors in this interpolation step of the adaptive process.

There are many factors which affect the efficiency of the adaptive refinement, such as the location of the split position in the split element, the order of the basis functions, the new position for moving a boundary in the case of lowest cost refinement. The frequency of adaptivity and the tolerance for terminating the adaptive process also affect the efficiency of the method. Therefore, an optimal strategy of adaptive process remains a key point to increasing efficiency. Development of such a strategy is under way but is not reported here.

4 Illustrations

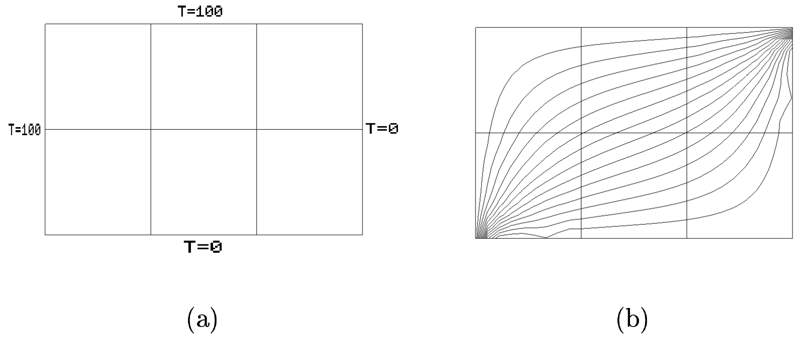
Heat Conduction

As mentioned previously, the Helmholtz/Poisson equation forms the core of our Navier-Stokes solver, hence, we first present a Poisson test problem in order to validate our method. The problem is one of two-dimensional heat conduction in a plate where the left and upper walls are held at temperature of 100 while the two other walls are held at temperature of 0 as shown in Fig. 2(a). The discontinuity of the two competing Dirichlet temperature boundary conditions at the upper right and lower left corners necessitates fine resolution since the spectral representation of a discontinuity leads to poor results. The initial solution was calculated with six equal-sized elements and is shown in Fig. 2. The unphysical kinks in temperature contour lines near the edges of the domain indicate that the numerical solution is poor. The adaptive method effected ten splits in both the upper right and lower left regions for a final solution using 26 elements (shown in Fig. 3(a)). The polynomial order in each element is four. The final solution shown in terms of contours of constant temperature in Fig. 3(b) is much improved. Error estimates (as defined in Section 1.3) have been reduced by approximately two orders of magnitude in all elements with exception for the smallest corner elements which contain the discontinuities.

Driven Cavity Stokes Flow

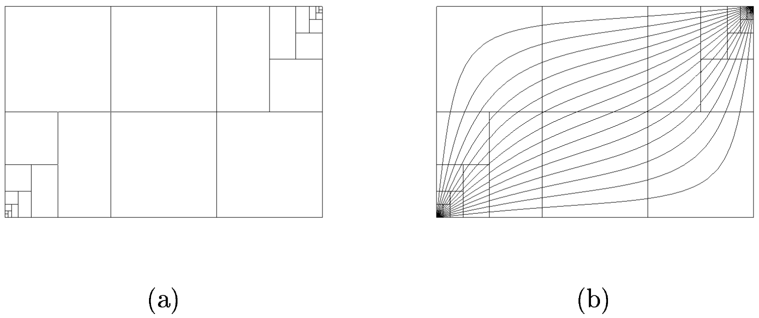
We now turn to a Stokes flow (where the nonlinear terms of the Navier-Stokes equations are negligible for very low Reynolds numbers): two-dimensional laminar flow in a cavity is used to illustrate how the adaptive process performs. This flow serves as a very popular test case (e.g., see references within [PT83]). There are discontinuities in velocity at the upper corners, where $u = 0$ from the no-slip condition on the vertical walls and $u = 1$ from the imposed driving flow on the upper wall. At these points there is a singularity as the vorticity becomes infinite. The driven cavity flow is solved adaptively starting with polynomials of order four on the four equal-sized element grid shown in Fig. 4(a). The Reynolds number for this case is 15. Fig. 5(a) presents the final mesh after a series of adaptation steps. Splitting of elements occurs around the upper corners where the singularities are. In Fig. 5(b), where the streamlines are plotted, we see the solution with 24 elements is well defined in most areas even near the singularities. Indeed, it is significantly improved over the nonadapted initial case shown in Fig. 4(b). The error estimate (as defined in Section 1.3) has been reduced to 10^{-2} in all elements except the smallest corner elements containing the singularities. These solutions are steady results calculated by an unsteady Stokes solver. The × points in the figures represent locations where the velocities were monitored in time to determine a steady result.

Figure 2 Heat conduction example before adaptive refinement: (a) elemental mesh (b) temperature contour lines



(a) (b)

Figure 3 Heat conduction example after adaptive refinement: (a) elemental mesh (b) temperature contour lines



(a) (b)

5 Conclusion

An adaptive spectral element method for the direct simulation of incompressible flows has been developed. The adaptive algorithm effectively diagnoses and refines regions of the flow where complexity of the solution requires increased resolution. The method has been demonstrated on two-dimensional examples in heat conduction and Stokes flows. The refinement has been limited to h-refinement for this paper. In the future, p-refinement will be combined with h-refinement for improved accuracy and efficiency.

Figure 4 Driven cavity flow before adaptive refinement: (a) elemental mesh (b) streamlines

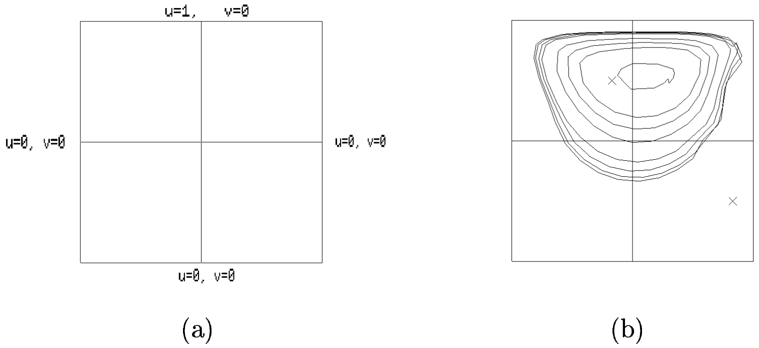
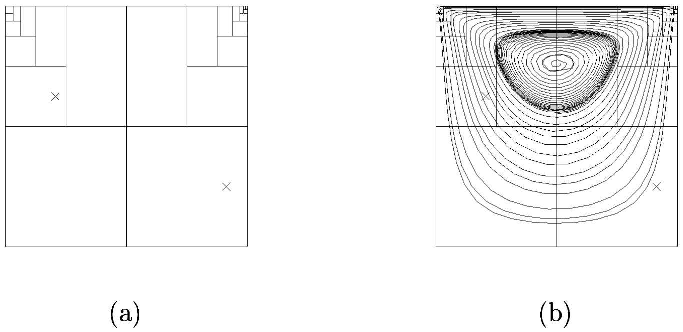


Figure 5 Driven cavity flow after adaptive refinement: (a) elemental mesh (b) streamlines



Acknowledgement

Support from the U. S. National Science Foundation under Grant ECS-9209347 is gratefully acknowledged.

REFERENCES

[FR94] Fischer P. and Rønquist E. (1994) Spectral element methods for large scale parallel Navier-Stokes calculations. *Computer Methods in Applied Mechanics and Engineering* 116: 69–76.

[Kar90] Karniadakis G. (1989/90) Spectral element simulations of laminar and turbulent flows in complex geometries. *Applied Numerical Mathematics* 6: 85–105.

[Mav90] Mavriplis C. (1990) A posteriori error estimators for adaptive spectral element techniques. In Wesseling P. (ed) *Notes on Numerical Fluid Mechanics*, volume 29, pages 333–342. Vieweg, Braunschweig.

[Mav94] Mavriplis C. (1994) Adaptive meshes for the spectral element method. *Computer Methods in Applied Mechanics and Engineering* 116: 77–86.

[MMP89] Maday Y., Mavriplis C., and Patera A. (1989) Nonconforming mortar element methods: Application to spectral discretization. In et al T. C. (ed) *Domain Decomposition Methods*. SIAM, Philadelphia.

[OK80] Orszag S. A. and Kells L. C. (1980) Transition to turbulence in plane Poiseuille and plane Couette flow. *Journal of Fluid Mechanics* 96: 159–205.

[Pat84] Patera A. (1984) A spectral element method for fluid dynamics: Laminar flow in a channel expansion. *Journal of Computational Physics* 54(3): 468–488.

[PT83] Peyret R. and Taylor T. (1983) *Computational Methods for Fluid Flow*. Springer, New York.

45

Optimized Krylov-Ventcell method. Application to convection-diffusion problems

Caroline Japhet

1 Introduction

In this paper a domain decomposition method with non-overlapping subdomains is presented, applied to the convection-diffusion problem:

$$\mathcal{L}(u) = cu + a(x,y)\frac{\partial u}{\partial x} + b(x,y)\frac{\partial u}{\partial y} - \nu\Delta u = f \text{ in } \Omega \qquad (1.1)$$
$$\mathcal{C}(u) = g, \text{ on } \partial\Omega$$

where Ω is a bounded open set of $\mathcal{R}^2$, $\boldsymbol{a} = (a,b)$ is the velocity field, ν is the viscosity, $\mathcal{C}$ is a linear operator, c is a constant which could be $c = \frac{1}{\Delta t}$ with Δt a time step of a backward-Euler scheme for solving the time dependent convection-diffusion problem. The strategy could be applied to other PDE's.

Substructuring formulation

Let $\bar{\Omega} = \cup_{i=1}^{N} \bar{\Omega}_i$, $\Omega_i \cap \Omega_j = \emptyset$, $i \neq j$. We denote by $\Gamma_{i,j}$ the common interface to Ω_i and Ω_j, $i \neq j$. The outward normal from Ω_i to Ω_j is $\boldsymbol{n}_{i,j}$ and $\boldsymbol{\tau}_{i,j}$ is a tangential unit vector.
The additive Schwarz algorithm [Lio89] is:

$$\begin{array}{rcl} \mathcal{L}(u_i^{n+1}) & = & f, \quad \text{in } \Omega_i \\ \mathcal{B}_{i,j}(u_i^{n+1}) & = & \mathcal{B}_{i,j}(u_j^n), \quad \text{on } \Gamma_{i,j},\ i \neq j \\ \mathcal{C}(u_i^{n+1}) & = & g, \quad \text{on } \partial\Omega_i \cap \partial\Omega \end{array}$$

Where $\mathcal{B}_{i,j}$ is an interface operator.
In [NRdS95], this algorithm is interpreted as a Jacobi algorithm applied to the interface problem

$$(Id - \mathcal{T})(H) = G \qquad (1.2)$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

where $\mathcal{T}$ is an interface operator, G a second member only depending on f and g. To accelerate convergence, the Jacobi algorithm is replaced by a BICGSTAB [Van92] or GMRES [SS86] algorithm.

In this paper, to accelerate convergence again, the interface conditions $\mathcal{B}_{i,j}$ between subdomains are chosen as partial differential operators of order 2 in the tangential direction to the interface, which minimize the rate of convergence of the Schwarz algorithm. We denote them by Optimized Order 2 conditions (OO2). To introduce them, other interface conditions are first recalled.

Interface conditions

General interface operators of order 2 in the tangential direction,

$$\mathcal{B}_{i,j} = \frac{\partial}{\partial \boldsymbol{n}_{i,j}} + c_1 + c_2 \frac{\partial}{\partial \boldsymbol{\tau}_{i,j}} + c_3 \frac{\partial^2}{\partial \boldsymbol{\tau}_{i,j}^2}, \quad \mathcal{B}_{j,i} = \frac{\partial}{\partial \boldsymbol{n}_{j,i}} + c_4 + c_5 \frac{\partial}{\partial \boldsymbol{\tau}_{j,i}} + c_6 \frac{\partial^2}{\partial \boldsymbol{\tau}_{j,i}^2} \quad (1.3)$$

where c_m are constants, $1 \leq m \leq 6$.

In [Des90],[CQ95], the interface conditions are of order 0 :

Taylor order 0 interface conditions (T0)

$$\mathcal{B}_{i,j} = \frac{\partial}{\partial \boldsymbol{n}_{i,j}} - \frac{\boldsymbol{a}.\boldsymbol{n}_{i,j} - \sqrt{(\boldsymbol{a}.\boldsymbol{n}_{i,j})^2 + 4c\nu}}{2\nu} \quad (1.4)$$

and $\mathcal{B}_{j,i}$ is defined as is $\mathcal{B}_{i,j}$, replacing $\boldsymbol{n}_{i,j}$ by $\boldsymbol{n}_{j,i}$.

In [NRdS95], [NR95], the interface conditions are of order 2 :

Taylor order 2 interface conditions (T2)

$$\begin{aligned}\mathcal{B}_{i,j} = \frac{\partial}{\partial \boldsymbol{n}_{i,j}} - \frac{\boldsymbol{a}.\boldsymbol{n}_{i,j} - \sqrt{(\boldsymbol{a}.\boldsymbol{n}_{i,j})^2 + 4c\nu}}{2\nu} + \frac{\boldsymbol{a}.\boldsymbol{\tau}_{i,j}}{\sqrt{(\boldsymbol{a}.\boldsymbol{n}_{i,j})^2 + 4c\nu}} \frac{\partial}{\partial \boldsymbol{\tau}_{i,j}} \\ - \frac{\nu}{\sqrt{(\boldsymbol{a}.\boldsymbol{n}_{i,j})^2 + 4c\nu}} (1 + \frac{(\boldsymbol{a}.\boldsymbol{\tau}_{i,j})^2}{(\boldsymbol{a}.\boldsymbol{n}_{i,j})^2 + 4c\nu}) \frac{\partial^2}{\partial \boldsymbol{\tau}_{i,j}^2}\end{aligned} \quad (1.5)$$

and $\mathcal{B}_{j,i}$ is defined as is $\mathcal{B}_{i,j}$, replacing $\boldsymbol{n}_{i,j}$ by $\boldsymbol{n}_{j,i}$ and $\boldsymbol{\tau}_{i,j}$ by $\boldsymbol{\tau}_{j,i}$.

Conditions (1.4), (1.5) can be seen as Taylor approximations of order 0 and order 2, for low wave numbers, of the artificial boundary conditions [EM77], [Hal86]: If $\Omega_i = \mathcal{R}^- \times \mathcal{R}$, $\Omega_j = \mathcal{R}^+ \times \mathcal{R}$, and $\Gamma_{i,j}$ is the axis $x = 0$, the artificial boundary conditions are $\partial_x - \Lambda^-$, $\partial_x - \Lambda^+$, with Λ^- the Dirichlet to Neumann operator of the right half plane defined as $\Lambda^- : u_0 \longrightarrow \frac{\partial w}{\partial x}(0, y)$ with w such as

$\mathcal{L}(w) = 0,\ x > 0,\ w(0, y) = u_0(y)$ at $x = 0$, and w bounded at infinity

The Dirichlet to Neumann operator of the left half plane Λ^+ is defined in the same way. When the coefficients of $\mathcal{L}$ are constant, if we denote by Λ_{ap}^+ and Λ_{ap}^- the Taylor approximations of order 0 or 2, for low wave numbers, of Λ^+ and Λ^-, they satisfy:

$$\Lambda_{ap}^+ + \Lambda_{ap}^- = \Lambda^+ + \Lambda^- = \frac{a}{\nu} \quad (1.6)$$

Then, $\mathcal{B}_{i,j} = \partial_x - \Lambda_{ap}^-$, $\mathcal{B}_{j,i} = \partial_x - \Lambda_{ap}^+$, and $\mathcal{B}_{j,i}$ can be obtained from $\mathcal{B}_{i,j}$, using (1.6). So in (1.3) the coefficients c_4, c_5, c_6 are obtained from c_1, c_2, c_3 (or reciprocally).

In [TB94], interface operators of order 1 are used : $c_3 = c_6 = 0$. The coefficients c_1, c_2, c_4, c_5 are chosen in order to minimize the convergence rate. As the minimization problem on the four parameters is very costly, an approximate minimization problem is solved, but it may lead to non convergence in some cases. The link between $\mathcal{B}_{i,j}$ and $\mathcal{B}_{j,i}$ as in (1.6) has not been done.

2 OO2 interface conditions

In this paper, the interface conditions are of order 2 as in (1.3) and are chosen as follows:

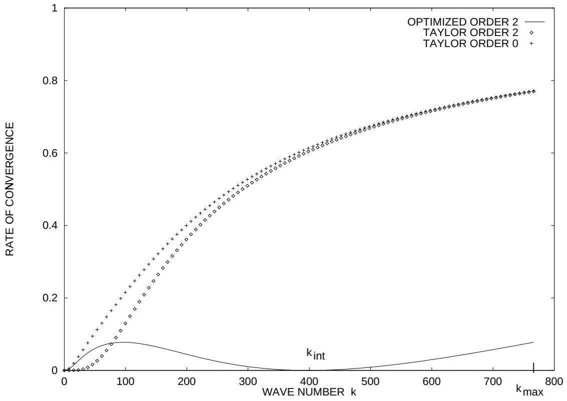
- First we link $\mathcal{B}_{i,j}$ and $\mathcal{B}_{j,i}$ as in (1.6). This means that c_4, c_5, c_6 are obtained from c_1, c_2, c_3:
$c_1 = c_1(\boldsymbol{a}.\boldsymbol{n}_{i,j}, \boldsymbol{a}.\boldsymbol{\tau}_{i,j})$, $c_2 = c_2(\boldsymbol{a}.\boldsymbol{n}_{i,j}, \boldsymbol{a}.\boldsymbol{\tau}_{i,j})$, $c_3 = c_3(\boldsymbol{a}.\boldsymbol{n}_{i,j}, \boldsymbol{a}.\boldsymbol{\tau}_{i,j})$, and $c_4 = c_1 + \frac{\boldsymbol{a}.\boldsymbol{n}_{i,j}}{\nu}$, $c_5 = c_2(\boldsymbol{a}.\boldsymbol{n}_{j,i}, \boldsymbol{a}.\boldsymbol{\tau}_{j,i})$, $c_6 = c_3(\boldsymbol{a}.\boldsymbol{n}_{j,i}, \boldsymbol{a}.\boldsymbol{\tau}_{j,i})$. So we only have to determine c_1, c_2, c_3.
- Then, we choose $c_1 = -\frac{\boldsymbol{a}.\boldsymbol{n}_{i,j} - \sqrt{(\boldsymbol{a}.\boldsymbol{n}_{i,j})^2 + 4c\nu}}{2\nu}$ so that the interface condition is exact for the lowest wave number.
- Finally, we compute c_2 and c_3 by minimizing the convergence rate of the Schwarz algorithm in the case of 2 subdomains and constant coefficients.

Advantages (N subdomains case): The minimization problem on c_2 and c_3 is on a set of conditions which verify (1.6), i.e. on a set of conditions which ensures (adding a condition on the sign of c_2) the convergence of the Schwarz algorithm (see [NN94]). So an approximate minimization problem on the same set of conditions will also ensure the convergence. In the case of 2 subdomains, the convergence is proved by computing explicitely the convergence rate. When the domain is decomposed in N subdomains (strips) the convergence rate is estimated in function of the convergence rate of the 2 subdomain case and the decomposition geometry. The convergence is proved by using techniques issued from formal language theory (see [NN94]).

The minimization problem on c_2 and c_3 is sought in term of wave numbers k: we minimize the maximum of the convergence rate function $k \to \rho(k, c_2, c_3)$ on the interval $|k| \leq k_{max}$ where k_{max} is a given constant, $k_{max} > 0$ (in the discrete case, $k_{max} = \frac{constant}{h}$ where h is the mesh size in y) (see [Jap96]).
The study of the function ρ leads us to determine only one parameter, which is a low wave number k_{int} (see Figure 1). This parameter is computed with a dichotomy algorithm, which is not costly. With k_{int} we can compute $c_2 = c_2(k_{int})$ and $c_3 = c_3(k_{int})$.

Theorem 2.1 *Let Ω be decomposed in 2 subdomains, with $a \in \mathcal{R}$, $a \neq 0$, $b = 0$ and $c \geq 0$ in (1.1). Let $k_{max} = \frac{\pi}{h}$ where h is the mesh size, and let $(\rho_{max})_{IC}$ be the maximum of ρ on $0 \leq k \leq k_{max}$ with the interface condition IC. Let $\alpha = 1 + \frac{4\nu c}{a^2}$. Then, when $h \to 0$:*
$(\rho_{max})_{T0} \approx 1 - \frac{2}{\pi}\alpha^{\frac{1}{2}}(\frac{|a|h}{\nu})$, $(\rho_{max})_{T2} \approx 1 - \frac{4}{\pi}\alpha^{\frac{1}{2}}(\frac{|a|h}{\nu})$,
$(\rho_{max})_{OO2} \approx 1 - 8\alpha^{\frac{1}{6}}(\frac{1}{4\pi}\frac{|a|h}{\nu})^{\frac{1}{3}}$

Figure 1 Rate of convergence versus wave numbers k, $\mathbf{0 \le k \le k_{max} = \frac{\pi}{h}}$
$a = 1,\ b = 1,\ \nu = 0.01,\ c = 0,\ h = \frac{1}{240}$



So the condition number is asymptotically much better for OO2 than for Taylor T0 or T2 interface conditions.

3 Numerical results

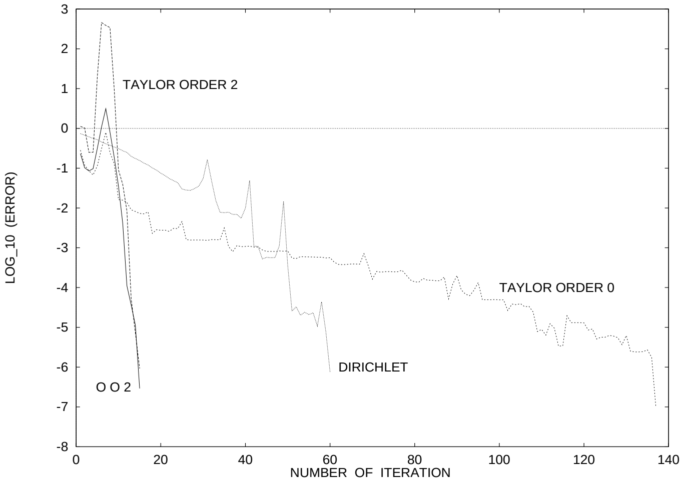
Let the problem be: $\mathcal{L}(u) = f, \quad 0 \le x \le 1,\ 0 \le y \le 1$
with $u(0, y) = \frac{\partial u}{\partial x}(1, y) = 0,\ 0 \le y \le 1,\ \frac{\partial u}{\partial y}(x, 1) = 0,\ u(x, 0) = 1,\ 0 \le x \le 1.$
We consider a rectangular finite difference grid with a mesh size h. The operator $\mathcal{L}$ and $\mathcal{B}_{i,j}$ are discretized by a standard upwind difference scheme (see [Fle90]). The unit square is decomposed into N rectangles with one overlapping mesh cell.
Remark: another discretization could be used, with a non-overlapping decomposition.
Algorithm :
The interface problem (1.2) is solved by a Bicgstab algorithm. This involves solving N independant subproblems which can be done in parallel. Each subproblem is solved by a direct method:
- First we compute the LU factorization of the matrix corresponding to the discretization of the subproblems. This is a parallel task.
- Then at each iteration of Bicgstab, we solve in parallel the subproblems using this LU factorization.
Important point: Each iteration has the same cost for all the interface conditions OO2,(1.4),(1.5),and Dirichlet, because the use of order 2 conditions does not increase the bandwidth of the local matrix.

We are more interested in the stationary case, so we take $c = 10^{-6}$ in the following results. The convergence is also significantly better when $c >> 1$.

The OO2 interface conditions give a significantly better convergence which is independant of the convection velocity angle to the interfaces (see Figure 2 and

Figure 3).

Figure 2 Error versus the number of iterations. Normal velocity to the interfaces, $\mathbf{16} \times \mathbf{1}$ subdomains, $a = y,\ b = 0,\ \nu = 0.01,\ c = 10^{-6},\ h = \frac{1}{240}$.



One of the advantages to have a convergence independant of the convection velocity angle to the interfaces is that for a given number of subdomains, the decomposition of the domain (in strips or in rectangles) doesn't affect the convergence (see Figures 2 and 4).

Figure 5 shows that the convergence with OO2 interface conditions is significantly better for a more general convection velocity (a rotating velocity) with a decomposition in 8 × 4 rectangles. The OO2 interface conditions are easy to implement and not cost increasing at each iteration. We observed numerically that the convergence with the OO2 interface conditions is also practically independant of the viscosity ν.
Moreover the OO2 interface conditions can be seen as a preconditioner for iterative methods, and the convergence for the studied numerical cases is independent of the mesh size (see Figures 6 and 7).

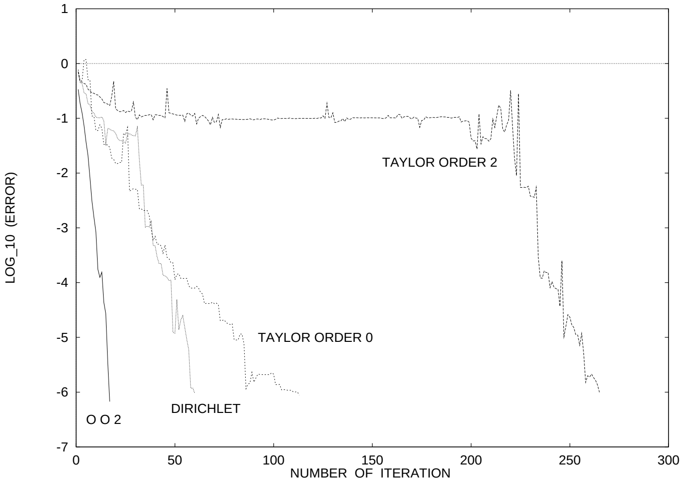
REFERENCES

[CQ95] Carlenzoli C. and Quarteroni A. (1995) Adaptive domain decomposition methods for advection-diffusion problems. In Babuska I. and Al. (eds) *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, number 75 in The IMA Volumes in Mathematics and its applications, pages 165–187. Springer-Verlag.

[Des90] Despres B. (1990) Décomposition de domaine et problème de Helmholtz. *C.R. Acad. Sci., Paris* 311(Série I): 313–316.

[EM77] Engquist B. and Majda A. (1977) Absorbing boundary conditions for the numerical simulation of waves. *Math. Comp.* 31(139): 629–651.

Figure 3 Error versus the number of iterations. Tangential velocity to the interfaces, **16 × 1** subdomains, $a = y,\ b = 0,\ \nu = 0.01,\ c = 10^{-6},\ h = \frac{1}{240}$.



[Fle90] Fletcher C. A. J. (1990) *Computational Techniques for Fluid Dynamics.* Springer Series in Computational Physics. Springer, second edition.

[Hal86] Halpern L. (1986) Artificial boundary conditions for the advection-diffusion equations. *Math. Comp.* 174: 425–438.

[Jap96] Japhet C. (1996) Optmisation des conditions d'interface pour les méthodes de décomposition de domaines; application à l'équation de convection-diffusion. Rapport interne, CMAP, Ecole Polytechnique. To appear.

[Lio89] Lions P. L. (1989) On the Schwarz alternating method 3: A variant for nonoverlapping subdomains. In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, SIAM, pages 202–223.

[NN94] Nataf F. and Nier F. (October 1994) Convergence rate of some domain decomposition methods for overlapping and nonoverlapping subdomains. Rapport interne 306, CMAP Ecole Polytechnique. Numerische Mathematik, in press.

[NR95] Nataf F. and Rogier F. (1995) Factorisation of the convection-diffusion operator and the Schwarz algorithm. *M^3AS, 5, n^1* pages 67–93.

[NRdS95] Nataf F., Rogier F., and de Sturler E. (1995) Domain decomposition methods for fluid dynamics. In Sequeira A. (ed) *Navier-Stokes equations on related non linear analysis*, pages 307–377. Plenum Press Corporation.

[SS86] Saad Y. and Schultz H. (1986) Gmres: Generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* 7: 856–869.

[TB94] Tan K. H. and Borsboom M. J. A. (1994) On generalized Schwarz coupling applied to advection-dominated problems. *Contemporary Mathematics* 180: 125–130.

[Van92] Van der Vorst H. A. (1992) Bi-cgstab: a fast and smoothly converging variant of bicg for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput* 13 n^0 2: 631–644.

Figure 4 Error versus the number of iterations.
4 × 4 subdomains, $a = y,\ b = 0,\ \nu = 0.01,\ c = 10^{-6},\ h = \frac{1}{240}$, with conditions of order 0 at cross points.

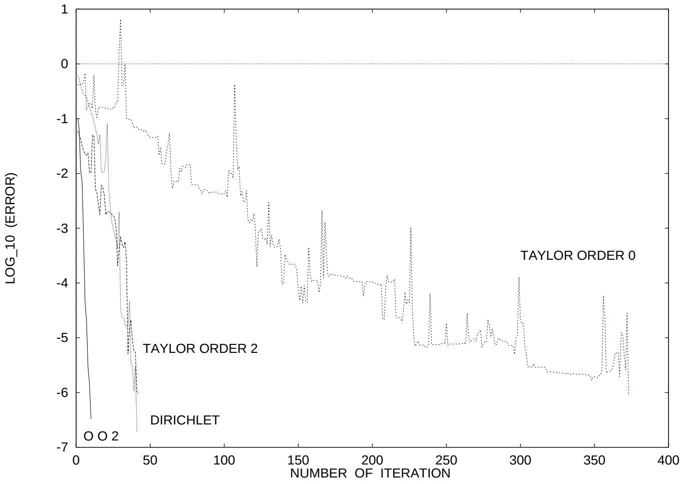


Figure 5 Error versus the number of iteration.
8 × 4 subdomains, $\nu = 0.01,\ c = 0,\ h = \frac{1}{240}$,
$a = -\sin(\pi(y - \frac{1}{2}))\cos(\pi(x - \frac{1}{2})),\ b = \cos(\pi(y - \frac{1}{2}))\sin(\pi(x - \frac{1}{2}))$, with conditions of order 0 at cross points.

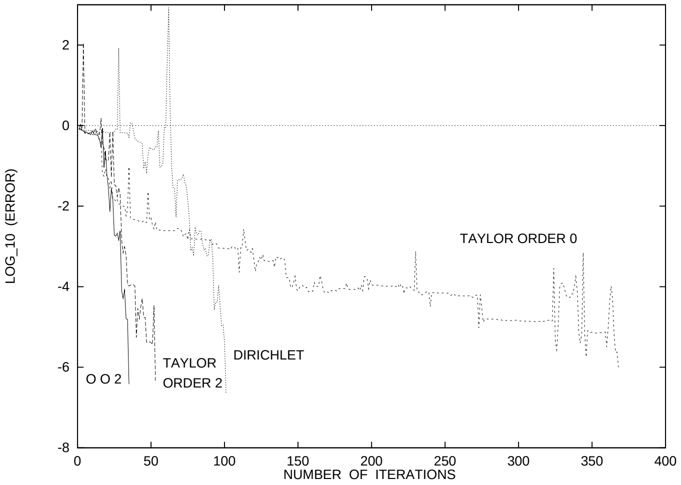


Figure 6 Number of iterations versus the mesh size.
16 × 1 subdomains, $a = y,\ b = 0,\ \nu = 0.01,\ c = 10^{-6},\ \max(Error) < 10^{-6}$

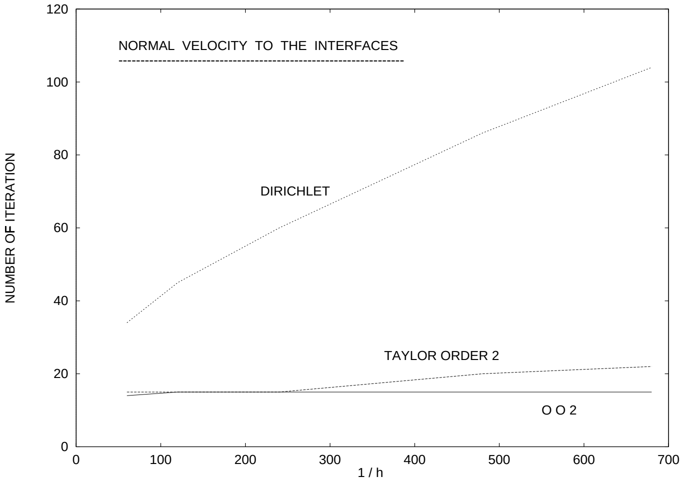
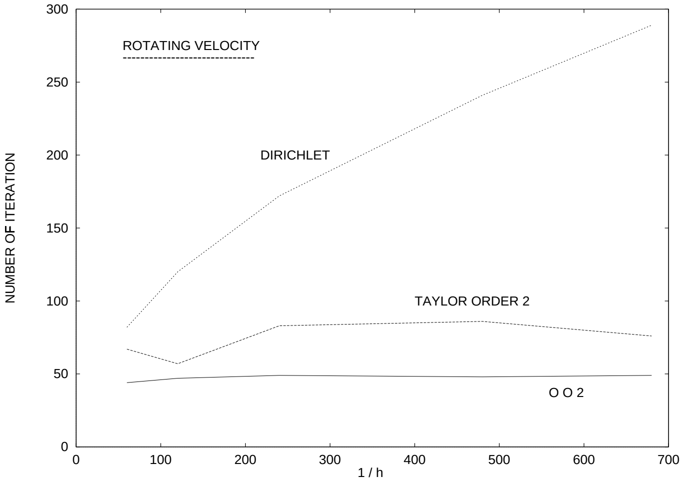


Figure 7 Number of iterations versus the mesh size.
16 × 1 subdomains, $\nu = 0.01,\ c = 10^{-6},\ \max(Error) < 10^{-6}$
$a = -\sin(\pi(y - \frac{1}{2}))\cos(\pi(x - \frac{1}{2})),\ b = \cos(\pi(y - \frac{1}{2}))\sin(\pi(x - \frac{1}{2}))$



46

Preconditioning of Two-dimensional Singular Integral Equations

Ke Chen

1 Introduction

Numerical solution of integral equations produces dense linear systems that give rise to unsymmetric matrices, in general. In the two-dimensional case, such systems can become too large for direct solution. Here we consider the application of conjugate gradient methods. For singular integral equations, such iterative methods require preconditioning for any convergence.

Preconditioning techniques proposed in the literature, involving sparse matrices, are mostly designed for one-dimensional integral equations, and based on considerations of efficiency. In [Che94] and [Che96], for 1D singular integral equations, we have given a theoretical justification for a class of preconditioners. Here we consider a generalization of this work to the 2D case. Such a theory is based on a suitable splitting of the underlying singular operator. Essentially the domain is divided into many subdomains in order to isolate singularities. Some experiments on Cauchy type bi-singular integral equations are reported. We have applied both the conjugate gradient normal method (CGN) and the generalized minimal residual method (GMRES), in connection with our proposed sparse preconditioners.

Our preliminary results show that the CGN with the proposed sparse preconditioners converges much faster than the GMRES. This conclusion is in agreement with our earlier work [Che96] on the one-dimensional case. The present work appears to be new as no extensive studies have been found in the literature regarding preconditioning bi-singular integral equations.

Our ultimate aim is to design efficient preconditioners for solving 2D singular boundary integral equations arising from 3D Helmholtz equations. This work has pointed out a way to achieve the aim.

To introduce our new work, we shall briefly describe how we decompose an integral operator in 1D based on domain splitting and further design sparse preconditioners.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 Decomposition of 1D Integral Operators

As in [Che96], denote an operator equation defined over interval $[a,b]$ by

$$\mathcal{A}\mathbf{u} = \mathcal{F},$$

and the corresponding discretized matrix equation by

$$Au = f.$$

The purpose of preconditioning is to choose a matrix M such that the linear system $MAu = Mf$ is more amenable to the use of iterative methods. As M or its inverse M^{-1} must be sparse for efficiency, we choose M from part of matrix A.

It turns out that, for singular integral equations, the following operator $\mathcal{D}_1$ contains all the singularity of dense operator $\mathcal{A}$,

$$\mathcal{D}_1 = \begin{pmatrix} \times & \times & & & & \times \\ \times & \times & \times & & & \\ & \times & \times & \times & & \\ & & \times & \times & \ddots & \\ & & & \ddots & \ddots & \times \\ \times & & & & \times & \times \end{pmatrix}.$$

That is, we have the decomposition $\mathcal{A} = \mathcal{D}_1 + \mathcal{C}_1$. Correspondingly the matrix decomposition will be $A = D_1 + C_1$. Then we take $M = D_1^{-1}$ as a preconditioner, where D_1 is of the same sparsity pattern as $\mathcal{D}_1$.

Depending on the type of numerical methods one intends to use, the following two preconditioners, derived from collocation methods collocating at nodes and at midpoints, respectively, have been found to be effective

$$\mathcal{D}_2 = \begin{pmatrix} \times & & & & & \times \\ \times & \times & & & & \\ & \times & \times & & & \\ & & \times & \times & & \\ & & & \ddots & \ddots & \\ & & & & \times & \times \end{pmatrix}, \qquad \mathcal{D}_3 = \begin{pmatrix} \times & & & & & \\ & \times & & & & \\ & & \times & & & \\ & & & \times & & \\ & & & & \ddots & \\ & & & & & \times \end{pmatrix}.$$

The three types of preconditioners above will be considered in the following sections. The generalization of other preconditioners is currently under investigation; see [Che96], [Vav92] and [Yan94].

3 A 2D Model Equation

As a first step to developing preconditioners for solving 2D singular boundary integral equations arising from 3D Helmholtz equations, we consider the following model integral equation

$$\int_a^b \int_c^e k(x,y;\xi,\eta)u(\xi,\eta)d\xi d\eta = f(x,y), \tag{3.1}$$

with singular kernel $k(x,y;\xi,\eta)$, i.e., $\mathcal{A}\mathbf{u} = \mathbf{F}$. Note that this model equation can represent the bi-singular integral equation arising from aerodynamics modelling ([Ell95]), when

$$k(x,y;\xi,\eta) = \frac{d(x,y;\xi,\eta)}{\pi^2(\xi - x)(\eta - y)},$$

where d denotes a smooth function.

4 Decomposition of 2D Singular Operators

Our main idea in designing sparse preconditioners is based on operator splittings. Bearing in mind that whenever (x,y) and (ξ,η) are distinct there is no singularity in the kernel, we shall consider a general case and two special cases of operator splittings. We assume that when (x,y) and (ξ,η) are on the opposite side of the boundary, there is no singularity. However, for boundary integral equations arising from 3D Helmholtz equations, boundary points coincide so the first part of our splittings should have a wrap-around structure as in the 1D case.

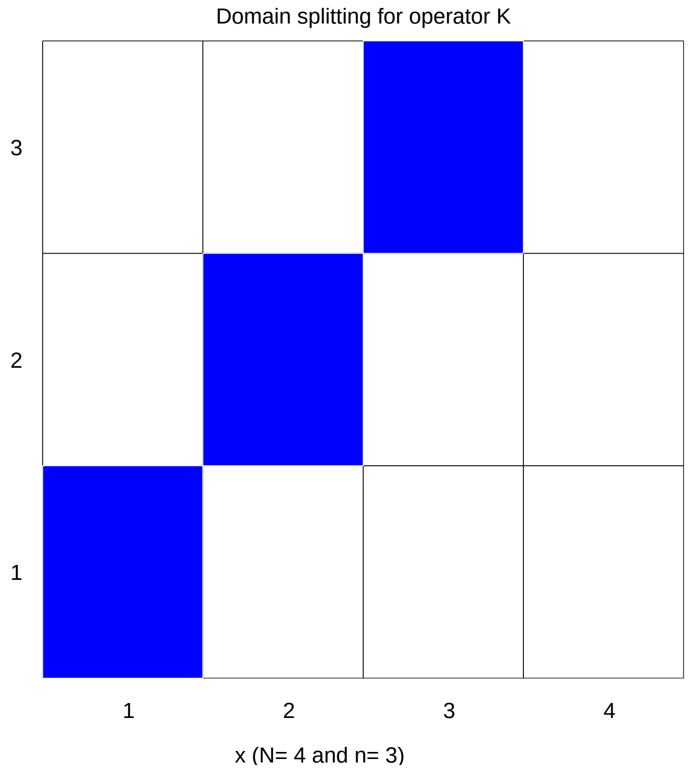
We now consider the decomposition of operator $\mathcal{A}$. To this end, partition the interval $[c,e]$ into n subintervals and $[a,b]$ into N subintervals. For simplicity, we shall take $n = 3$ and $N = 4$ in following discussions; see Fig. 1.

Then operator $\mathcal{A}$ can be written in a matrix form

$$\left[\begin{array}{cccc|cccc|cccc}
\mathcal{A}_{11}^{11} & \mathcal{A}_{12}^{11} & \mathcal{A}_{13}^{11} & \mathcal{A}_{1N}^{11} & \mathcal{A}_{21}^{11} & \mathcal{A}_{22}^{11} & \mathcal{A}_{23}^{11} & \mathcal{A}_{2N}^{11} & \mathcal{A}_{n1}^{11} & \mathcal{A}_{n2}^{11} & \mathcal{A}_{n3}^{11} & \mathcal{A}_{nN}^{11} \\
\mathcal{A}_{11}^{12} & \mathcal{A}_{12}^{12} & \mathcal{A}_{13}^{12} & \mathcal{A}_{1N}^{12} & \mathcal{A}_{21}^{12} & \mathcal{A}_{22}^{12} & \mathcal{A}_{23}^{12} & \mathcal{A}_{2N}^{12} & \mathcal{A}_{n1}^{12} & \mathcal{A}_{n2}^{12} & \mathcal{A}_{n3}^{12} & \mathcal{A}_{nN}^{12} \\
\mathcal{A}_{11}^{13} & \mathcal{A}_{12}^{13} & \mathcal{A}_{13}^{13} & \mathcal{A}_{1N}^{13} & \mathcal{A}_{21}^{13} & \mathcal{A}_{22}^{13} & \mathcal{A}_{23}^{13} & \mathcal{A}_{2N}^{13} & \mathcal{A}_{n1}^{13} & \mathcal{A}_{n2}^{13} & \mathcal{A}_{n3}^{13} & \mathcal{A}_{nN}^{13} \\
\mathcal{A}_{11}^{1N} & \mathcal{A}_{12}^{1N} & \mathcal{A}_{13}^{1N} & \mathcal{A}_{1N}^{1N} & \mathcal{A}_{21}^{1N} & \mathcal{A}_{22}^{1N} & \mathcal{A}_{23}^{1N} & \mathcal{A}_{2N}^{1N} & \mathcal{A}_{n1}^{1N} & \mathcal{A}_{n2}^{1N} & \mathcal{A}_{n3}^{1N} & \mathcal{A}_{nN}^{1N} \\
\hline
\mathcal{A}_{11}^{21} & \mathcal{A}_{12}^{21} & \mathcal{A}_{13}^{21} & \mathcal{A}_{1N}^{21} & \mathcal{A}_{21}^{21} & \mathcal{A}_{22}^{21} & \mathcal{A}_{23}^{21} & \mathcal{A}_{2N}^{21} & \mathcal{A}_{n1}^{21} & \mathcal{A}_{n2}^{21} & \mathcal{A}_{n3}^{21} & \mathcal{A}_{nN}^{21} \\
\mathcal{A}_{11}^{22} & \mathcal{A}_{12}^{22} & \mathcal{A}_{13}^{22} & \mathcal{A}_{1N}^{22} & \mathcal{A}_{21}^{22} & \mathcal{A}_{22}^{22} & \mathcal{A}_{23}^{22} & \mathcal{A}_{2N}^{22} & \mathcal{A}_{n1}^{22} & \mathcal{A}_{n2}^{22} & \mathcal{A}_{n3}^{22} & \mathcal{A}_{nN}^{22} \\
\mathcal{A}_{11}^{23} & \mathcal{A}_{12}^{23} & \mathcal{A}_{13}^{23} & \mathcal{A}_{1N}^{23} & \mathcal{A}_{21}^{23} & \mathcal{A}_{22}^{23} & \mathcal{A}_{23}^{23} & \mathcal{A}_{2N}^{23} & \mathcal{A}_{n1}^{23} & \mathcal{A}_{n2}^{23} & \mathcal{A}_{n3}^{23} & \mathcal{A}_{nN}^{23} \\
\mathcal{A}_{11}^{2N} & \mathcal{A}_{12}^{2N} & \mathcal{A}_{13}^{2N} & \mathcal{A}_{1N}^{2N} & \mathcal{A}_{21}^{2N} & \mathcal{A}_{22}^{2N} & \mathcal{A}_{23}^{2N} & \mathcal{A}_{2N}^{2N} & \mathcal{A}_{n1}^{2N} & \mathcal{A}_{n2}^{2N} & \mathcal{A}_{n3}^{2N} & \mathcal{A}_{nN}^{2N} \\
\hline
\mathcal{A}_{11}^{n1} & \mathcal{A}_{12}^{n1} & \mathcal{A}_{13}^{n1} & \mathcal{A}_{1N}^{n1} & \mathcal{A}_{21}^{n1} & \mathcal{A}_{22}^{n1} & \mathcal{A}_{23}^{n1} & \mathcal{A}_{2N}^{n1} & \mathcal{A}_{n1}^{n1} & \mathcal{A}_{n2}^{n1} & \mathcal{A}_{n3}^{n1} & \mathcal{A}_{nN}^{n1} \\
\mathcal{A}_{11}^{n2} & \mathcal{A}_{12}^{n2} & \mathcal{A}_{13}^{n2} & \mathcal{A}_{1N}^{n2} & \mathcal{A}_{21}^{n2} & \mathcal{A}_{22}^{n2} & \mathcal{A}_{23}^{n2} & \mathcal{A}_{2N}^{n2} & \mathcal{A}_{n1}^{n2} & \mathcal{A}_{n2}^{n2} & \mathcal{A}_{n3}^{n2} & \mathcal{A}_{nN}^{n2} \\
\mathcal{A}_{11}^{n3} & \mathcal{A}_{12}^{n3} & \mathcal{A}_{13}^{n3} & \mathcal{A}_{1N}^{n3} & \mathcal{A}_{21}^{n3} & \mathcal{A}_{22}^{n3} & \mathcal{A}_{23}^{n3} & \mathcal{A}_{2N}^{n3} & \mathcal{A}_{n1}^{n3} & \mathcal{A}_{n2}^{n3} & \mathcal{A}_{n3}^{n3} & \mathcal{A}_{nN}^{n3} \\
\mathcal{A}_{11}^{nN} & \mathcal{A}_{12}^{nN} & \mathcal{A}_{13}^{nN} & \mathcal{A}_{1N}^{nN} & \mathcal{A}_{21}^{nN} & \mathcal{A}_{22}^{nN} & \mathcal{A}_{23}^{nN} & \mathcal{A}_{2N}^{nN} & \mathcal{A}_{n1}^{nN} & \mathcal{A}_{n2}^{nN} & \mathcal{A}_{n3}^{nN} & \mathcal{A}_{nN}^{nN}
\end{array}\right].$$

In the general case, we obtain the splitting $\mathcal{A} = \mathcal{D}_1 + \mathcal{C}_1$, where all singularity of $\mathcal{A}$ is contained in a $\mathcal{D}_1$ that has a block tridiagonal structure, i.e., represented from the

Figure 1 The domain splitting for a general operator $\mathcal{A}$ ($n = 3$ & $N = 4$)

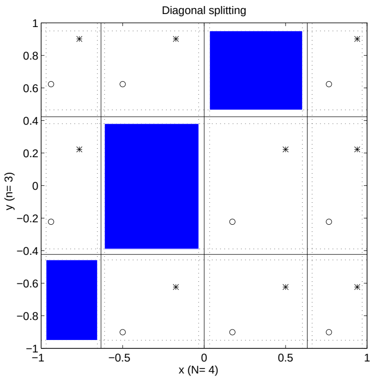


nonzero entries below (boxed and un-boxed)

$$\left[\begin{array}{cccc|cccc|cccc}
\boxed{\times} & \times & & & \times & \times & & & & & & \\
\boxed{\times} & \boxed{\times} & \times & & \times & \times & \times & & & & & \\
& \boxed{\times} & \boxed{\times} & \times & & \times & \times & \times & & & & \\
& & \boxed{\times} & \boxed{\times} & & & \times & \times & & & & \\
\hline
\boxed{\times} & \times & & & \boxed{\times} & \times & & & \times & \times & & \\
\boxed{\times} & \boxed{\times} & \times & & \boxed{\times} & \boxed{\times} & \times & & \times & \times & \times & \\
& \boxed{\times} & \boxed{\times} & \times & & \boxed{\times} & \boxed{\times} & \times & & \times & \times & \times \\
& & \boxed{\times} & \boxed{\times} & & & \boxed{\times} & \boxed{\times} & & & \times & \times \\
\hline
& & & & \boxed{\times} & \times & & & \boxed{\times} & \times & & \\
& & & & \boxed{\times} & \boxed{\times} & \times & & \boxed{\times} & \boxed{\times} & \times & \\
& & & & & \boxed{\times} & \boxed{\times} & \times & & \boxed{\times} & \boxed{\times} & \times \\
& & & & & & \boxed{\times} & \boxed{\times} & & & \boxed{\times} & \boxed{\times}
\end{array}\right]. \tag{4.2}$$

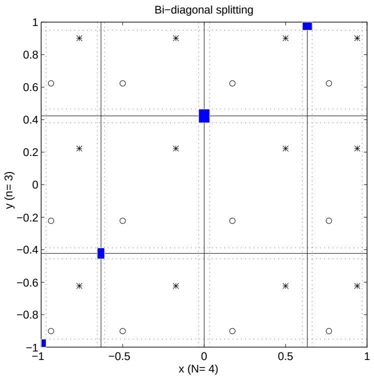
To simplify $\mathcal{D}_1$ further, we first consider only part of the underlying domain that surrounds all nodes; see Fig. 2. This corresponds to numerical methods collocating at nodes. We obtain the splitting $\mathcal{A} = \mathcal{D}_2 + \mathcal{C}_2$, where all singularities of $\mathcal{A}$ are contained in $\mathcal{D}_2$, which has a block bi-diagonal structure, e.g., represented by the boxed entries in (4.2).

Figure 2 The domain splitting to generate a bi-diagonal splitting of $\mathcal{A}$



We then consider the opposite part of the domain that excludes all nodes; see Fig. 3. This is associated with numerical methods collocating at interior points. Now the operator is split as $\mathcal{A} = \mathcal{D}_3 + \mathcal{C}_3$, where all singularities of $\mathcal{A}$ are contained in $\mathcal{D}_3$, which has a diagonal structure, e.g., the diagonal in (4.2).

Figure 3 The domain splitting to generate a diagonal splitting of $\mathcal{A}$



In summary, all three operator splitting strategies discussed are such that the preconditioned equation has a compact operator and therefore ideal spectral properties. At the matrix level, we propose three sparse block preconditioners: tri-diagonal ($M = M_1$), bi-diagonal ($M = M_2$), and diagonal ($M = M_3$) matrices.

Table 1 Convergence results of CGN

n	Nodes n^2	Unpreconditioned	Tri-diagonal M_1	Bi-diagonal M_2	Diagonal M_3
4	16	12	13	12	10
8	64	88	33	51	23
16	256	*	88	215	41
32	1024	*	216	972	68

5 Numerical Results

We now present some numerical results of using the above described sparse preconditioners for solving

$$\frac{1}{\pi}\int_{-1}^{1}\frac{1}{\pi}\int_{-1}^{1}\frac{d(x,y;\xi,\eta)}{(\xi-x)(\eta-y)}u(\xi,\eta)d\xi d\eta = f(x,y),$$

with Kutta conditions $\left\{\begin{array}{ll} u(1,y)=0 & -1<y<1, \\ u(x,1)=0 & -1<x<1. \end{array}\right.$

Write the index 0 solution as

$$u(x,y)=\sqrt{\frac{1-x}{1+x}}\sqrt{\frac{1-y}{1+y}}\Psi(x,y)$$

to give the equation

$$\frac{1}{\pi}\int_{-1}^{1}\sqrt{\frac{1-\xi}{1+\xi}}\frac{d\xi}{\xi-x}\left(\frac{1}{\pi}\int_{-1}^{1}\sqrt{\frac{1-\eta}{1+\eta}}\frac{\Psi(\xi,\eta)d\eta}{\eta-y}\right) = f(x,y).$$

Taking $N=n$, an appropriate discretization is given by

$$\sum_{j=1}^{n}\sum_{k=1}^{n}\frac{d(\eta_{rn},\eta_{sn};\xi_{jn},\xi_{kn})}{(\xi_{jn}-\eta_{rn})(\xi_{kn}-\eta_{sn})}\frac{(1-\xi_{jn})(1-\xi_{kn})}{(n+1/2)(n+1/2)}\Psi_{nn}(\xi_{jn},\xi_{kn}) = f(\eta_{rn},\eta_{sn})$$

where

$$\left\{\begin{array}{ll} \xi_{jn}=\cos\left(\frac{2j\pi}{2n+1}\right), & j=1,\cdots,n \\ \eta_{jn}=\cos\left(\frac{(2j-1)\pi}{2n+1}\right), & j=1,\cdots,n. \end{array}\right.$$

We consider $\Psi(x,y)=1$, $d(x,y;\xi,\eta)=1$, and $f(x,y)=1$. In Tables 1 and 2, we show the number of steps required to reduce the residual error to below a tolerance of $10^{(-2-\log(n)/\log(2))}=10^{-(\ell+2)}$ for $n=2^\ell$, where '*' denotes no convergence or the iteration steps exceed $3n/2$, CGN stands for the conjugate gradient normal method and GMRES(m) for the generalized minimal residual method; see [NRT92]. This particular

Table 2 Convergence results of re-started GMRES(3)

n	Nodes n^2	Unpreconditioned	Tri-diagonal M_1	Bi-diagonal M_2	Diagonal M_3
4	16	2	6	12	*
8	64	2	22	26	*
16	256	2	*	53	*
32	1024	*	*	198	*

choice of tolerance ensures that the residual is of a comparable magnitude to the truncation error that would result with a direct solver.

We can observe that CGN produces results as predicted and M_3 is the best preconditioner because the numerical method collocates at interior points (see Fig. 3). However the performance of GMRES is somewhat erratic; we have experimented with an increased m and observed similar results. Here each step of GMRES involves m matrix-vector multiplications while CGN involves two matrix-vector multiplications. Therefore our preliminary results suggest that CGN is suitable for bi-singular integral equations with operator splitting based sparse preconditioners (M_3 and M_1). For integral equations, in general, CGN with suitable preconditioners may out perform other iterative solvers; see [Che97] for some discussion.

Acknowledgement

The author wishes to thank Professor David Elliott for suggesting the test problem and its numerical solution.

REFERENCES

[Che94] Chen K. (1994) Efficient iterative solution of linear systems from discretizing singular integral equation. *Elec. Tran. Numer. Anal.* 2: 76–91.

[Che96] Chen K. (1996) Solution of singular boundary element equations based on domain splitting. In Glowinski R., Périaux J., Shi Z.-C., and Widlund O. B. (eds) *Proc. Eighth Int. Conf. on Domain Decomposition Meths.* Wiley and Sons, Chichester.

[Che97] Chen K. (1997) On preconditioning techniques for dense linear systems from bundary elements. *to appear* .

[Ell95] Elliott D. (1995) Private communication. Maths Department, University of Tasmania, Australia.

[NRT92] Nachtigal N., Reddy S. C., and Trefethen L. N. (1992) How fast are nonsymmetric matrix iterations. *SIAM J. Matrix Anal. Appl.* 13(3): 778–795.

[Vav92] Vavasis S. (1992) Preconditioning for boundary integral equations. *SIAM J. Matrix. Anal. Appl.* 13(3): 905–925.

[Yan94] Yan Y. (1994) Sparse preconditioned iterative methods for dense linear systems. *SIAM J. Sci. Comp.* 15(5): 1190–1200.

47

Parallel Preconditioners for a Fourth-order Discretization of the Viscous Bürgers Equation

Samuel Kortas and Philippe Angot

1 Introduction

We present a parallel implementation of a new high precision conservative scheme "CFV4" that uses additive Schwarz domain decomposition methods to precondition three Krylov solvers. Parallel performance results are given for the Cray T3D. The additional use of a multigrid regular finite volume solver on each subdomain accelerates the observed elapsed times and appears optimal as soon as the number of unknowns in each subdomain is sufficient.

2 The Fourth-order Compact Conservative Scheme CFV4

We solve the 2D nonlinear unsteady viscous Bürgers equation in the bounded unit square domain $\Omega_0 = [0,1] \times [0,1]$ in the time interval $[0,T]$:

$$
\begin{aligned}
\frac{\partial u}{\partial t}(\mathbf{x},t) + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} - \nabla \cdot (\nu_x \nabla u)(\mathbf{x},t) &= f_u(\mathbf{x},t) \quad \text{for } (\mathbf{x},t) \in \Omega_0 \times [0,T] \\
\frac{\partial v}{\partial t}(\mathbf{x},t) + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} - \nabla \cdot (\nu_y \nabla v)(\mathbf{x},t) &= f_v(\mathbf{x},t) \\
u|_{\partial\Omega_0}(\mathbf{x},t) = g_u(\mathbf{x},t) \quad & u(\mathbf{x},0) = u_0(\mathbf{x}) \\
v|_{\partial\Omega_0}(\mathbf{x},t) = g_v(\mathbf{x},t) \quad & v(\mathbf{x},0) = v_0(\mathbf{x})
\end{aligned}
\tag{2.1}
$$

where u_0, v_0, f_u, f_v, g_u, g_v are sufficiently regular functions and ν_x and ν_y are some given functions of space and time, possibly non-smooth. The 2D domain Ω_0 is meshed by uniform grid with $\Delta x = \Delta y = h$ and $[0,T]$ is divided into time steps Δt.

Because our final application is the unsteady incompressible Navier-Stokes system, we only consider divergence-free solutions satisfying (2.1). For these, the conservative

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

4^{th}-order time discretized formulation of (2.1) is integrated on a control volume $\mathcal{V}$ and the advective terms are linearized in time as follows:

$$\int_{\mathcal{V}} \frac{\Phi(u^{n+1},...,u^{n-3})}{12\Delta t} dv + \int_{\partial\mathcal{V}} u^m(u^*,v^*)^T \cdot \ \mathbf{n} d\sigma - \int_{\partial\mathcal{V}} \nu_x \nabla u^{n+1} \cdot \ \mathbf{n} d\sigma = \int_{\mathcal{V}} f_u^{n+1} dv \quad (2.2)$$

$$\int_{\mathcal{V}} \frac{\Phi(v^{n+1},..,v^{n-3})}{12\Delta t} dv + \int_{\partial\mathcal{V}} v^m(u^*,v^*)^T \cdot \ \mathbf{n} d\sigma - \int_{\partial\mathcal{V}} \nu_y \nabla v^{n+1} \cdot \ \mathbf{n} d\sigma = \int_{\mathcal{V}} f_v^{n+1} dv \quad (2.3)$$

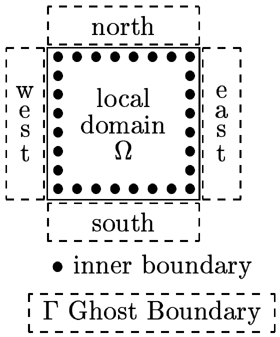
where $\Phi(\psi^{n+1},...,\psi^{n-3}) = 25\psi^{n+1} - 48\psi^n + 36\psi^{n-1} - 16\psi^{n-2} + 3\psi^{n-3}$ is chosen in order to obtain a 4^{th}-order discretization of $\frac{\partial\psi}{\partial t}$ at $(n+1)\Delta t$ and $\psi^* = 4\psi^n - 6\psi^{n-1} + 4\psi^{n-2} - \psi^{n-3}$ is the consistent 4^{th}-order Richardson extrapolation of ψ^{n+1}. We obtain two 4^{th}-order accurate time discretizations, whether the advective terms are treated implicitly $(m = n+1)$ or explicitly $(m = *)$.

Figure 1 Block structure of $\mathbf{A}_u$

$$\mathbf{A}_u = \begin{pmatrix} DDDPPPPP \\ DDDDPPPP \\ DDDDDPPP \\ PDDDDDPP \\ PPDDDDDP \\ PPPDDDDD \\ PPPPDDDD \\ PPPPPDDD \end{pmatrix}$$

with $\begin{cases} \text{D: Dense Matrix} \\ \text{P: Pentadiagonal Matrix} \end{cases}$

Figure 2 Ghost-cells and the local domain



In this conservative formulation, we need to evaluate the fluxes of the advective and diffusive terms at the surface $\partial\mathcal{V}$ of $\mathcal{V}$ with a 4^{th}-order spatial consistency. To do so, we derive linear "compact-like" relations [Lel92] linking discrete values of (u,v) defined at the center of each control volume on one line or column of the mesh, with their first derivatives taken at the center of the horizontal and vertical interfaces of these control volumes $\mathcal{V}$.

We obtain a matricial expression $\mathbf{A}_u.\ \mathbf{u}^{n+1}$, where $\mathbf{u}^{n+1}$ gathers all the u-unknowns at $(n+1)\Delta t$ stored in lexicographical order. Because the nonsymmetric and rather dense-profile matrix $\mathbf{A}_u$ (Figure 1) is expensive to build explicitly, we prefer to evaluate $\mathbf{A}_u.\ \mathbf{u}$ by calculating the solution of tridiagonal systems: 2 along each row and 2 along each column. For each point, we also need to perform four 5-point-stencil discrete 1D integration and one 5 × 5 point-stencil discrete 2D integration. Similar relations hold for v. Full details on the calculation of $\mathbf{A}.\ \mathbf{U} = (\mathbf{A}_u.\ \mathbf{u}, \mathbf{A}_v.\ \mathbf{v})^T$ are given in [KA96a] and the efficient parallel resolution of tridiagonal systems distributed over a row or column of processors is discussed in [KA96b].

3 Parallel Schwarz-Krylov Solvers

Parallel Partitioned Solvers

Because $\mathbf{A}$ is nonsymmetric, we implement some Krylov-type solvers to solve $\mathbf{A}.\ \mathbf{U} =\ \mathbf{f}$ at each time step: the straightforward implementation of BiCGSTAB and BiCGSTAB(2) solvers [Van95] on a MIMD machine, and the partitioned BiCGSTAB and GMRESR(k) methods [VV91] preconditioned by an additive Schwarz method.

Thanks to the use of "stencils of communication" and global reduction operations, these algorithms are implemented in a pleasant and easy-to-read form particularly well-suited to overlapping or nonoverlapping domain decomposition methods. As shown on Figure 2, each local subdomain Ω is extended with a ghost-cell boundary that contains either duplicate values of grid points known by the immediate neighbor processors or values set by the discretization of the boundary conditions on $\partial\Omega_0$.

A call to REFRESH_NEWS($\mathbf{u}$,$\mathcal{S}_{Default}$) duplicates the neighbor values of $\mathbf{u}$ contained in adjacent processors in the four cardinal directions (N,E,W,S). All operations preceded by GLOBAL_ are performed on each node before calling a global native reduction routine that sends back the global result to all nodes. A complete description of the implementation and performance of this model of programming on IBM SP2, Cray T3D, and IPSC/i860 can be found in [KA96b] or [AKF96].

Figure 3 Parallel BiCGSTAB

```
SET $\mathbf{x}_0(= \mathbf{u}^*$ or $\mathbf{u}^*_{rec})$ , $\epsilon(= 10^{-8})$, $\epsilon_{stop}$
REFRESH_NEWS( $\mathbf{x}_0$,$\mathcal{S}_{Default}$)
 $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\ \mathbf{x}_0$, $\widehat{\mathbf{r}_0} = \mathbf{r}_0$, $\mathbf{v} = \mathbf{p} = 0$
$\rho_0 = \alpha = \omega = 1$
WHILE $\frac{\text{GLOBAL\_}\|\ \mathbf{r}_k\|_2}{\text{GLOBAL\_}\|\ \mathbf{r}_0\|_2} > \epsilon$
    and GLOBAL_$\|\ \mathbf{r}_k\|_2 > \epsilon_{stop}$ DO
        $\rho =$ GLOBAL_( $\widehat{\mathbf{r}_0}^T\ \mathbf{r}$)
        $\beta = \alpha\rho/\rho_0\omega$, $\rho_0 = \rho$
        $\mathbf{p} = \mathbf{r} + \beta(\ \mathbf{p} - \omega\ \mathbf{v})$
        SOLVE $\mathbf{M}\widehat{\mathbf{p}} = \mathbf{p}$
        REFRESH_NEWS($\widehat{\mathbf{p}}$,$\mathcal{S}_{Default}$)
        $\mathbf{v} = \mathbf{A}\widehat{\mathbf{p}}$
        $\alpha = \rho_1/$ GLOBAL_($\widehat{\mathbf{r}_0}^T\ \mathbf{v}$)
        $\mathbf{s} = \mathbf{r} - \alpha\ \mathbf{v}$
        SOLVE $\mathbf{M}\ \mathbf{z} = \mathbf{s}$
        REFRESH_NEWS( $\mathbf{z}$,$\mathcal{S}_{Default}$)
        $\mathbf{t} = \mathbf{A}\ \mathbf{z}$
        $\omega = \frac{\text{GLOBAL\_( } \mathbf{t}^T\ \mathbf{s})}{\text{GLOBAL\_( } \mathbf{t}^T\ \mathbf{t})}$
        $\mathbf{x} = \mathbf{x} + \alpha\widehat{\mathbf{p}} + \omega\ \mathbf{z}$
        $\mathbf{r} = \mathbf{s} - \omega\ \mathbf{t}$
ENDWHILE
```

Figure 4 Parallel GMRESR(k)

```
SET $\mathbf{x}_0(= \mathbf{u}^*$ or $\mathbf{u}^*_{rec})$, $\epsilon(= 10^{-8})$, $\epsilon_{stop}$
REFRESH_NEWS( $\mathbf{x}_0$,$\mathcal{S}_{Default}$)
 $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\ \mathbf{x}_0$
$k = 0$
WHILE $\frac{\text{GLOBAL\_}\|\ \mathbf{r}_k\|_2}{\text{GLOBAL\_}\|\ \mathbf{r}_0\|_2} > \epsilon$
    and GLOBAL_$\|\ \mathbf{r}_k\|_2 > \epsilon_{stop}$ DO
        $k = k + 1$
        SOLVE $\mathbf{M}_k\ \mathbf{v}_k^{(1)} = \mathbf{r}_{k-1}$
        REFRESH_NEWS( $\mathbf{v}_k^{(1)}$,$\mathcal{S}_{Default}$)
        $\mathbf{c}_k^{(1)} = \mathbf{A}\ \mathbf{v}_k^{(1)}$
        FOR $i = 1, \cdots, k - 1$ DO
            $\alpha_i =$ GLOBAL_( $\mathbf{c}_i^T\ \mathbf{c}_k^{(i)}$)
            $\mathbf{c}_k^{(i+1)} = \mathbf{c}_k^{(i)} - \alpha_i\ \mathbf{c}_i$
            $\mathbf{v}_k^{(i+1)} = \mathbf{v}_k^{(i)} - \alpha_i\ \mathbf{v}_i$
        $\mathbf{c}_k = \mathbf{c}_k^{(k)}/$ GLOBAL_ $\|\ \mathbf{c}_k^{(k)}\|_2$
        $\mathbf{v}_k = \mathbf{v}_k^{(k)}/$ GLOBAL_ $\|\ \mathbf{c}_k^{(k)}\|_2$
        $\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{v}_k$ GLOBAL_( $\mathbf{c}_k^T\ \mathbf{r}_{k-1}$)
        $\mathbf{r}_k = \mathbf{r}_{k-1} - \mathbf{c}_k$ GLOBAL_( $\mathbf{c}_k^T\ \mathbf{r}_{k-1}$)
ENDWHILE
```

Parallel Preconditioners

Figure 3 shows the parallel version of a preconditioned BiCGSTAB algorithm. If $\mathbf{M} = \mathbf{I}$, we obtain a "natural" parallel partitioned implementation of this solver: **Part_BiCGSTAB**. Similarly, we implement **Part_BiCGSTAB(2)**.

Following [Meu91], [Le 94] or [Ang94], we precondition these solvers by an additive Schwarz domain decomposition method to solve at each iteration $\mathbf{M}\widehat{\mathbf{p}} = \mathbf{p}$ or $\mathbf{M}\ \mathbf{z} = \mathbf{s}$. In this preconditioning step, we first extend the local contribution of $\mathbf{s}$ or $\mathbf{p}$ to the extended overlapping local subdomain with calls to communication stencil routines. On each subdomain handled by a different node, we solve "exactly" a local problem, similar to (2.2–2.3) taken this time with homogeneous Dirichlet boundary condition. A BiCGSTAB(2) solver is used to reduce the local residual by eight orders of magnitude. During this step, no communication occurs. No coarse grid solver is implemented. The global solution vector $\widehat{\mathbf{p}}$ or $\mathbf{z}$ is finally rebuilt from the projections of the solution of each local problem.

By keeping the same 4^{th}-order accurate spatial discretization for the local problems than for the initial one, we obtain a "classical" Krylov-Schwarz additive solver: **DDM4(l)_BiCGSTAB** where $\delta = l.h$ is half of the overlap within subdomains. As shown on Figure 4, we also implement a GMRESR solver preconditioned as above: **DDM4(l)_GMRESR(k)**.

We can also discretize and solve the local problems thanks to classical second-order finite volumes. These algorithms – **DDM2(l)_BiCGSTAB** and **DDM2(l)_GMRESR(k)** – are eventually accelerated by using a multigrid solver instead of BiCGSTAB(2). In parallel on each subdomain, and totally independently, V-Cycles are carried out on a hierarchy of grids. We perform 3 pre- and 2 post-smoothing Gauss-Seidel iterations per level, and we solve the 8×8 coarsest problem by BiCGSTAB to obtain the solvers: **DDM2MG(l)_BiCGSTAB** and **DDM2MG(l)_GMRESR(k)** (for the test problem studied here, $k = 15$ avoids restarting the GMRES algorithm).

We also test the efficiency of a reconjugation technique combined with the GMRESR-type solvers. Inspired by [Rou95], we store the "best" descent directions $(\mathbf{c}_{k\_rec}, \mathbf{v}_{k\_rec}) = (\mathbf{c}_k, \mathbf{v}_k)$ satisfying a bound on $\text{GLOBAL\_}(\mathbf{c}_k^T\ \mathbf{r}_{k-1})$. In the following time steps, these directions are reused to start the iteration process from a better initial guess $\mathbf{x}_0 = \mathbf{u}^*_{rec} = \mathbf{u}^* + \mathbf{v}_{k\_rec}\ \text{GLOBAL\_}(\mathbf{c}^T_{k\_rec}(\mathbf{b} - \mathbf{A}\ \mathbf{u}^*))$.

4 Convergence Results

As detailed in [KA96a], fourth-order accuracy in time and space is reached with such a discretization. In the following, we only present results on a test solution for a dipole diagonally crossing Ω_0 for $t \in [0, T = 1]$: $\big(u(x,y,t) = 10\ (t-y)\, e^{\frac{-(t-x)^2-(t-y)^2}{\nu}}, v(x,y,t) = 10\ (x-t)\, e^{\frac{-(t-x)^2-(t-y)^2}{\nu}}\big)$. All the tests are run with diffusive and advective terms both treated implicitly (i.e., $m = n+1$ in 2.2-2.3). Results when the advective terms are treated explicitly are collected in [Kor97].

Table 1 displays the convergence rate $\rho_{10} = \sqrt[N]{\frac{\| \mathbf{r}_N \|_2}{\| \mathbf{r}_0 \|_2}}$ observed after ten time steps for the test solution solved on a 256×256 mesh with $\Delta t = 1.25 \times 10^{-2}$,

Table 1 Convergence Rate for 256 × 256 Dipole Problem with no Richardson Extrapolation ($\epsilon = 10^{-8}$, $\epsilon_{stop} = 10^{-4}$)

Algorithm	2 dom.	4 dom.	8 dom.	16 dom.	32 dom.	64 dom.
Part_BiCGSTAB	5.70e-1	5.70e-1	5.80e-1	5.80e-1	5.80e-1	5.80e-1
Part_BiCGSTAB(2)	2.20e-1	2.20e-1	2.20e-1	2.10e-1	2.30e-1	2.20e-1
DDM4(2)_BiCGSTAB	-	-	-	1.60e-2	2.00e-2	1.90e-2
DDM4(4)_BiCGSTAB	-	-	7.80e-4	8.30e-4	1.40e-3	1.70e-3
DDM4(8)_BiCGSTAB	-	1.30e-5	6.60e-5	2.30e-5	2.50e-5	3.60e-5
DDM2(2)_BiCGSTAB	na	3.60e-2	3.70e-2	3.60e-2	4.50e-2	4.90e-2
DDM2(4)_BiCGSTAB	6.70e-3	9.80e-3	9.80e-3	1.00e-2	1.20e-2	1.20e-2
DDM2(8)_BiCGSTAB	6.70e-3	7.40e-3	6.40e-3	6.60e-3	8.00e-3	9.00e-3
DDM2MG(8)_BiCGSTAB	5.90e-3	7.60e-3	1.30e-2	1.40e-2	1.40e-2	1.80e-2
DDM4(2)_GMRESR	-	-	9.20e-2	9.40e-2	1.10e-1	1.20e-1
DDM4(4)_GMRESR	-	-	2.60e-2	2.70e-2	2.90e-2	3.30e-2
DDM4(8)_GMRESR	-	2.40e-3	2.50e-3	2.70e-3	3.20e-3	4.30e-3
DDM2(2)_GMRESR	1.30e-1	1.70e-1	1.70e-1	1.70e-1	1.90e-1	2.00e-1
DDM2(4)_GMRESR	7.10e-2	8.40e-2	8.50e-2	8.70e-2	9.40e-2	1.00e-1
DDM2(8)_GMRESR	5.10e-2	5.40e-2	5.60e-2	5.70e-2	6.30e-2	6.80e-2
DDM2MG(8)_GMRESR	na	7.20e-2	1.00e-1	1.00e-1	1.00e-1	9.50e-2

$\nu = 10^{-2}$, $\epsilon = 10^{-8}$, $\epsilon_{stop} = 10^{-4}$, and N the number of iterations needed to meet this convergence criteria ($N > 10$). No Richardson extrapolation is used and no reconjugation technique is activated to ameliorate the initial guess (i.e., $\mathbf{x}_0 = \mathbf{u}^n$). As awaited for a DDM-based method, the convergence rate appears better when the overlap δ increases, and worse when the number of involved subdomains increases. The better rate observed in the case of **DDM4_BiCGSTAB** solvers comes from the preconditioning step called twice for this algorithm instead of just once for the **DDM4_GMRESR** solvers.

Table 2 illustrates a classical result: if the overlap δ is kept proportional to the size of the local subdomains H, the number of iterations needed for a domain decomposition preconditioned solver is asymptotically bounded independently of h. With no preconditioner, one asymptotically needs $O(1/h)$ iterations as shown for **Part_BiCGSTAB**. Other calculations not presented here also show independence in H/h.

So far, we have not investigated the need for a coarse grid solver but a forthcoming article will further examine the "scaling" behavior of the convergence rates.

5 Parallel Performance Results

Even if the convergence rate is better, the elapsed times of an algorithm on up-to-date parallel computers can vary widely, being highly sensitive to the workload compared to the amount of communication needed to run an algorithm on several processors. Domain decomposition techniques are therefore particularly well-adapted to distributed computing because of the high data locality of the preconditioning step. We present here timing results measured on the Cray T3D.

In Table 3, we first observe a certain hierarchy among the solvers tested in Parallel.

Table 2 Behavior of the different algorithms when the relative overlap $\frac{\delta}{H}$ is kept constant. Test made on 64 Cray T3D nodes, to calculate one time step ($\epsilon = 10^{-16}$ $\epsilon_{stop} = 10^{-20}$)

Algorithm	global mesh	local mesh	l	# iterations	time(s)
Part_BiCGSTAB	128× 128	16× 16	-	34	1
Part_BiCGSTAB	256× 256	32× 32	-	75	8
Part_BiCGSTAB	384× 384	48× 48	-	107	23
Part_BiCGSTAB	512× 512	64× 64	-	147	55
Part_BiCGSTAB	640× 640	80× 80	-	204	117
Part_BiCGSTAB	768× 768	96× 96	-	223	181
DDM4(l)_BiCGSTAB	128× 128	16× 16	1	15	15
DDM4(l)_BiCGSTAB	256× 256	32× 32	2	10	73
DDM4(l)_BiCGSTAB	384× 384	48× 48	3	12	323
DDM4(l)_BiCGSTAB	512× 512	64× 64	4	9	537
DDM4(l)_BiCGSTAB	640× 640	80× 80	5	10	1233
DDM4(l)_BiCGSTAB	768× 768	96× 96	6	-	-
DDM2(l)_BiCGSTAB	128× 128	16× 16	1	16	3
DDM2(l)_BiCGSTAB	256× 256	32× 32	2	13	20
DDM2(l)_BiCGSTAB	384× 384	48× 48	3	12	60
DDM2(l)_BiCGSTAB	512× 512	64× 64	4	11	129
DDM2(l)_BiCGSTAB	640× 640	80× 80	5	11	249
DDM2(l)_BiCGSTAB	768× 768	96× 96	6	11	418
Part_BiCGSTAB	120× 120	15× 15	-	32	2
Part_BiCGSTAB	240× 240	30× 30	-	64	7
Part_BiCGSTAB	480× 480	60× 60	-	129	42
Part_BiCGSTAB	960× 960	120× 120	-	319	400
DDM2MG(l)_BiCGSTAB	120× 120	15× 15	1	16	3
DDM2MG(l)_BiCGSTAB	240× 240	30× 30	2	14	17
DDM2MG(l)_BiCGSTAB	480× 480	60× 60	4	14	86
DDM2MG(l)_BiCGSTAB	960× 960	120× 120	8	14	194

Remark: In tabulated results, a dash "-" means that the calculation was not pursued because of a predicted high elapsed time of computation. An "na" means that the result is not available.

Table 3 Elapsed time (s) to perform 10 time steps for a 256 × 256 Problem on Cray T3D with no Richardson Extrapolation ($\epsilon = 10^{-8}$, $\epsilon_{stop} = 10^{-4}$)

Algorithm	2 dom.	4 dom.	8 dom.	16 dom.	32 dom.	64 dom.
Part_BiCGSTAB	629	307	158	78	46	25
Part_BiCGSTAB(2)	582	293	153	78	44	25
DDM4(2)_BiCGSTAB	-	-	-	1432	778	396
DDM4(4)_BiCGSTAB	-	-	1745	909	519	255
DDM4(8)_BiCGSTAB	-	2155	1239	711	435	245
DDM2(2)_BiCGSTAB	-	775	400	208	106	56
DDM2(4)_BiCGSTAB	833	655	352	190	105	57
DDM2(8)_BiCGSTAB	873	666	359	218	125	80
DDM2MG(8)_BiCGSTAB	628	327	177	86	49	26
DDM4(2)_GMRESR	-	-	-	1190	685	329
DDM4(4)_GMRESR	-	-	1704	890	513	258
DDM4(8)_GMRESR	-	2151	1244	712	443	247
DDM2(2)_GMRESR	1059	739	379	197	108	57
DDM2(4)_GMRESR	845	552	295	159	85	52
DDM2(8)_GMRESR	771	506	291	168	99	59
DDM2MG(8)_GMRESR	na	267	166	73	41	22

For all purely partitioned solvers, **Part_BiCGSTAB(2)** and **Part_BiCGSTAB** show the same efficiency and score surprisingly well. In comparison, the "classical" domain decomposition preconditioning technique performs poorly: in all configurations of overlap and amount of nodes, **Part_BiCGSTAB** and **Part_BiCGSTAB(2)** greatly exceed **DDM4(l)_BiCGSTAB** and **DDM4(l)_GMRESR** in term of elapsed time. With a 4^{th}-order discretization, the better rate of convergence obtained thanks to DDM-preconditioning doesn't counterbalance the overload of computation demanded to solve each local problem quasi-exactly.

One can significantly improve this disappointing result with a conventional second-order finite volume solver combined with a multigrid acceleration technique. As shown on Tables 4 and 3, the resolution of the local problems discretized at the order 2 scores better than classical DDM4-preconditioned solver. Only a multigrid acceleration of this DDM2-solver equals or performs better than **Part_BiCGSTAB** or **Part_BiCGSTAB(2)**. Table 2 clearly tilts in favor of a multigrid second order preconditioner as soon as the amount of local points is sufficiently important to achieve a decent acceleration (194s vs 400s for the 960 × 960 mesh).

The influence of the size of the overlapping δ exhibits different optimal values for each algorithm. However, for the Krylov additive Schwarz solvers taken with an optimal overlapping, the GMRESR-based solver always appears slightly faster than BiCGSTAB-global solvers. It may come from the different number of synchronization points present in both algorithms: they are more numerous in GMRESR but localized in the orthogonalization loop whereas they are spread out all over the algorithm for BiCGSTAB. The collected global reduction operations are not indeed as penalizing as the alternation of stencil calls.

Finally, if compared with Table 3, Table 4 underlines the important influence of the Richardson extrapolation taken as initial guess: for the same stopping criteria ϵ_{stop},

Table 4 Elapsed time observed (s) to perform 10 time steps for a 256 × 256 Dipole Problem on Cray T3D with Richardson Extrapolation ($\epsilon = 10^{-8}$ and $\epsilon_{stop} = 10^{-4}$)

Algorithm	2 dom.	4 dom.	8 dom.	16 dom.	32 dom.	64 dom.
Part_BiCGSTAB	455	226	119	59	33	19
Part_BiCGSTAB(2)	467	230	123	60	35	19
DDM4(8)_BiCGSTAB	-	2183	1269	719	447	246
DDM2(8)_BiCGSTAB	683	432	240	144	84	48
DDM2MG(8)_BiCGSTAB	411	238	146	74	40	22
DDM2(8)_GMRESR	553	370	212	123	78	50
DDM2MG(8)_GMRESR	na	205	128	53	30	16
DDM4(8)_GMRESR/Reconj	-	1649	946	542	330	185
DDM2(8)_GMRESR/Reconj	568	360	207	119	72	43
DDM2MG(8)_GMRESR/Reconj	na	190	117	55	30	17

then observed elapsed time highly decreases for all tested algorithms. One can also note the insignificant impact of the reconjugation technique.

6 Acknowledgement

The authors gratefully acknowledge use of the Cray T3D located at the C.E.A. Grenoble and I.D.R.I.S., and of the IBM SP2 at C.N.U.S.C. The research of the first author is supported by the CEA-CISI-CNRS BDI scholarship contract #740 384 CEA/DAM/CEL-V #3633.

REFERENCES

[AKF96] Angot P., Kortas S., and Fürst J. (1996) Parallel and distributed multi-domain methods for numerical fluid dynamics. In Bubak M. and Mościński J. (eds) *High Performance Computing in Europe on IBM Platforms, Sup'Eur 96 Conf. Proc.*, pages 111–120. ACC CYFRONET, Kraków.

[Ang94] Angot P. (1994) Parallel multi-level and domain decomposition methods. *Calculateurs Parallèles, L.T.C.P.* 6: 9–14.

[KA96a] Kortas S. and Angot P. (1996) A new class of high order compact finite volume schemes. *Numerical Methods for PDEs* To submit.

[KA96b] Kortas S. and Angot P. (June 1996) A practical and portable model of programming for iterative solvers on distributed memory machines. *Parallel Computing* 22: 487–512.

[Kor97] Kortas S. (1997) *Résolution Haute Précision des équations de Navier-Stokes sur machines parallèles à mémoire distribuée.* PhD dissertation in applied mathematics, Université de Provence, C.M.I. in progress.

[Le 94] Le Tallec P. (1994) Domain decomposition methods in computational mechanics. *Computational Mechanics Advances* 1: 121–220.

[Lel92] Lele S. K. (1992) Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics* 103: 16–42.

[Meu91] Meurant G. A. (1991) Numerical experiment with a domain decomposition method for parabolic problems on parallel computers. In Glowinski R., Kuznetsov Y. A., Meurant G. A., Périaux J., and Widlund O. B. (eds) *Proc. Fourth Int. Conf.*

on Domain Decomposition Meths., chapter 32, pages 394–408. SIAM, Philadelphia.

[Rou95] Roux F.-X. (1995) Parallel implementation of a domain decomposition method for non-linear elasticity problems. In Keyes D. E., Saad Y., and Truhlar D. G. (eds) *Domain-Based Parallelism and Problem Decomposition Methods in Computational Sciences and Engineering*, chapter 10, pages 161–175. SIAM, Philadelphia.

[SBG96] Smith B., Bjørstad P., and Gropp W. (1996) *Domain Decomposition, Parallel Multilevel Methods for Elliptic Differential Equations.* CAMBRIDGE University Press.

[Van95] Van der Vorst H. A. (May 1995) Parallel iterative solution methods for linear systems arising from discretized PDE's. Lecture Notes on Parallel Iterative Methods for discretized PDE's. AGARD Special Course on Parallel Computing in CFD, available from http://www.math.ruu.nl/people/vorst/#lec.

[VV91] Van der Vorst H. A. and Vuik C. (1991) GMRESR: A family of nested GMRES methods. Technical Report DUT-TWI-91-80, Delft University of Technology, Department of Technical Mathematics and Informatics, Delft, The Netherlands. avalaible from ftp://ftp.twi.tudelft.nl/TWI/publications/tech-reports/1991/DUT-TWI-91-80.ps.gz.

48

Iterative Substructuring Preconditioners for the Mortar Finite Element Method

Catherine Lacour

1 The Mortar Element Method

The mortar element method, first introduced by C.Bernardi, Y.Maday and A.T.Patera in [BMP90], has the advantage of allowing non-matching nonoverlapping grids at the interfaces between subdomains. Therefore, the method permits implementation of different approximations on each subdomain, which means that grids can be built completely independently. It is designed to provide an efficient parallelizable evaluation and solution framework. In our case, we use the finite element method for each subdomain. In [RLJK96], we describe more fully the space of Lagrange multipliers chosen.

Let us consider the model elliptic problem in Ω: Find u in $H_0^1(\Omega)$ such that

$$\forall v \in H_0^1(\Omega), \int_\Omega \nabla u . \nabla v \; dx + c \int_\Omega uv \; dx = \int_\Omega fv \; dx \qquad (1.1)$$

This formulation is very handy for introducing nonoverlapping domain decompositions. Indeed, assume that Ω is partitioned into nonoverlapping (Lipschitz) subdomains $\overline{\Omega} = \bigcup_{k=1}^K \overline{\Omega}^k, \quad \Omega^k \cap \Omega^\ell = \emptyset$ if $k \neq \ell$

Problem (1.1) can be rewritten as follows: *Find $u \in H_0^1(\Omega)$ such that*

$$\forall v \in H_0^1(\Omega), \quad \sum_{k=1}^K \int_{\Omega^k} \nabla(u_{|\Omega^k}) \nabla(v_{|\Omega^k}) dx + c \sum_{k=1}^K \int_{\Omega^k} u_{|\Omega^k} v_{|\Omega^k} \; dx = \sum_{k=1}^K \int_{\Omega^k} f_{|\Omega^k} v_{|\Omega^k} dx$$

Instead of searching an element u defined globally over Ω, it is more convenient, especially when local discretizations are to be used, to search for a K-uple $u^* = (u_1, \ldots, u_K)$. The space V spanned by these restrictions

$$V = \{v^* = (v_1, \ldots, v_K), \quad \exists v \in H_0^1(\Omega), \forall k, 1 \leq k \leq K, \; v_k = v_{|\Omega^k}\}$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

can be conveniently rewritten as an aggregate of the local spaces

$$X_k = \{v_k \in H^1(\Omega^k), \quad v_k = 0 \text{ over } \partial\Omega^k \cap \partial\Omega\}$$

as follows

$$V^* = \{v^* = (v_1, \dots, v_K) \in \Pi_{k=1}^K X_k, \quad \forall k, \ell,\ 1 \leq k, \ell \leq K,\ v_k = v_\ell \text{ over } \partial\Omega^k \cap \partial\Omega^\ell\}.$$

This leads naturally to introduce the notation $\Gamma_{k,\ell} = \partial\Omega^k \cap \partial\Omega^\ell$. The constraint across the interface $\Gamma_{k,\ell}$ can be relaxed by inducing the definition of a Lagrange multiplier in the Euler equation. The Lagrange multiplier belongs to a closed subspace M of $\Pi_{1 \leq k < \ell \leq K} H^{-1/2}(\Gamma_{k,\ell})$. The problem (1.1) is equivalent to the following one : *Find $u^* \in V^*$ such that*

$$\forall v^* \in V^*, \quad \sum_{k=1}^K \int_{\Omega^k} \nabla u_k \nabla v_k dx + c \sum_{k=1}^K \int_{\Omega^k} u_k v_k dx = \sum_{k=1}^K \int_{\Omega^k} f_k v_k dx \tag{1.2}$$

Discretization

We discretize the problem by the Galerkin method. Let us consider a parameter h standing for a discretization parameter. For any value of h, for any k, $1 \leq k \leq K$, we introduce a finite dimensional subspace X_h^k of $X_k \cap C^0(\overline{\Omega}^k)$. For any k, $1 \leq k \leq K$, $\Gamma^{k,j}$, $1 \leq j \leq j(k)$ stand for the (eventually curved) segments which coincide with the edges of Ω^k, ($j(k)$ denote the number of edges of Ω^k). We then define the skeleton S as the union of all edges of all subdomains: $S = \bigcup_{k=1}^K \bigcup_{j=1}^{j(k)} \overline{\Gamma}^{k,j}$. Finally, we choose a finite set $\mathcal{M}$ of pairs $m = (k,j)$ such that the $\Gamma^{k,j}$ are disjoint from each other. We denote by γ^m, and we call mortars, these $\Gamma^{k,j}$. To describe the discrete space, we begin by defining trace spaces.

- First, for any k, $1 \leq k \leq K$ and for any j, $1 \leq j \leq j(k)$, we set $W_h^{k,j}$

$$W_h^{k,j} = \{v_{|\Gamma^{k,j}}, v \in X_h^k\} \quad .$$

- Next, for any $m^* = (k,j)$ not in $\mathcal{M}$, we choose a space $\tilde{W}_h^m$ of discrete functions on the non-mortar sides. The product of all these spaces provides a global discretization $\tilde{W}_h$ of the functions on the skeleton S by $\tilde{W}_h = \Pi_{m \notin \mathcal{M}} \tilde{W}_h^m$.

For any $m \in \mathcal{M}$, we denote by W_h^m the space $W_h^{k(m),j(m)}$. The mortar space is defined by $W_h = \{';'_{|\gamma^m} \in W_h^m, m \in \mathcal{M}\}$. The discrete space X_h is the space of functions v_h on Ω such that:

- For any k, $1 \leq k \leq K$, $v_{h,k} = v_{h|\Omega^k} \in X_h^k$.
- there exists a function $' \in W_h$ such that:
  If $\Gamma^{k,j}$ is a mortar, $v_{h,k|\Gamma^{k,j}} = '$
  If $\Gamma^{k,j}$ is not a mortar

$$\forall \psi \in \tilde{W}_h^{k,j}, \int_{\Gamma^{k,j}} (v_{h,k|\Gamma^{k,j}} - ')\psi \, d\tau \quad = \quad 0 \quad .$$

The discretized variational formulation is: *Find $u_h \in X_h$ such that*

$$\forall v_h \in X_h, \ \sum_{k=1}^{K} \int_{\Omega^k} \nabla u_{h,k}.\nabla v_{h,k} \ dx + c \sum_{k=1}^{K} \int_{\Omega^k} u_{h,k} v_{h,k} \ dx = \sum_{k=1}^{K} \int_{\Omega^k} f v_{h,k} \ dx \quad . \tag{1.3}$$

The problem can be reformulated into a saddle point problem.
Let a_h be the symmetric bilinear form on $X_h \times X_h$:

$$a_h(u_h, v_h) = \sum_{k=1}^{K} \int_{\Omega^k} \nabla u_{h,k}.\nabla v_{h,k} \ dx + c \sum_{k=1}^{K} \int_{\Omega^k} u_{h,k} v_{h,k} \ dx \quad ,$$

and b_h the bilinear form on $X_h \times \tilde{W}_h$:

$$b_h(v_h, \mu_h) = \sum_{1 \le k < \ell \le K} \int_{\Gamma_{k,\ell}} (v_{h,k} - v_{h,\ell}) \mu_h \quad .$$

We can associate to a_h the linear operator A_h and to b_h the linear operator B_h such that $a_h(u_h, v_h) = (A_h u_h, v_h)$ and $b_h(v_h, \mu_h) = (B_h v_h, \mu_h)$. Therefore, the problem (1.3) admits a following saddle-point formulation: Find the pair (u_h, λ_h) in $X_h \times \tilde{W}_h$ such that

$$\begin{aligned} A_h u_h + B_h^t \lambda_h &= f_h \\ B_h u_h &= 0 \quad . \end{aligned} \tag{1.4}$$

2 Extension of the Dual Schur Method Preconditioner

Conforming Case

Let us consider a subdomain Ω_i. We number its degrees of freedom beginning by those lying inside Ω_i and finishing by those lying on the interfaces between Ω_i and the others subdomains. With this numbering, the stiffness matrix of Ω_i has the following block representation:

$$A_i = \begin{pmatrix} A_{ii} & A_{if} \\ A_{fi} & A_{ff} \end{pmatrix} \tag{2.5}$$

The restriction of the matrix B on interface is the matrix which makes the correspondence between the degrees of freedom on interface and the degrees of freedom of the Lagrange multipliers.
The dual operator on each subdomain is given by $D^{(i)} = B_i A_i^{-1} B_i^t$

The interpretation of the preconditioner is to find a matrix M which is a good approximation of D^{-1} so as to apply the conjugate gradient method on a well conditioned problem. The preconditioner chosen in [Rou89] is $\bar{M} = \sum_i B_i A_i B_i^t$.

Nonconforming Case

The interface matrix is given by the bilinear form associated to the trace operator seen in (1.4).

The interface matrix is written as $B_i = P_i R_i$ where P_i is a projection matrix and R_i is the restriction matrix on the interface.

Therefore, the dual operator matrix D is given by

$$D = \sum_i B_i A_i^+ B_i^t = \sum_i P_i R_i A_i^+ R_i^t P_i^t.$$

The preconditioner chosen for the nonconforming case is

$$M2 = \sum_i (P_i P_i^t)^{-1} (P_i R_i A_i R_i^t P_i^t)(P_i P_i^t)^{-1}$$

where $(P_i P_i^t)^{-1}$ are matrix terms.

3 Hierarchical Basis of the Lagrange Multipliers Space

Our motivation comes from the work of H. Yserentant, [Yse86]. In his paper, the condition number of the stiffness matrices arising in the discretization of selfadjoint and positive definite elliptic problems by finite element methods when using hierarchical basis of the finite element spaces instead of the usual nodal bases is analysed. It is showed in [BDY88] that the condition number of such a stiffness matrix behaves like $0((\log K)^2)$ where K is the condition number of the stiffness matrix with respect to a nodal basis. In case of a triangulation with uniform mesh size h this means that the stiffness matrix with respect to a hierarchical basis has a condition number behaving like $0((\log \frac{1}{h})^2)$ instead of $0((\frac{1}{h})^2)$ for a nodal basis.

Therefore, in the same idea, we consider the dual operator matrix with respect to a hierarchical basis not of the finite element space but of the Lagrange multipliers space.

We begin in this section by stating the basic methodology for the building of a hierarchical basis. We start with a coarse initial mesh $\mathcal{T}_1$. Beginning with this mesh, we construct a nested family $\{\mathcal{T}_\alpha\}$ of meshes. In this section, $\mathcal{T}_{\alpha+1}$ is obtained from $\mathcal{T}_\alpha$ by subdividing any elements of the mesh $\mathcal{T}_\alpha$.

The space $V(\Omega)$ is approached by the succession of the finite element spaces corresponding $\{V_\alpha(\Omega)\}$. $V_{\alpha+1}$ is obtained from V_α by adding the basis functions $\Phi_A^{\alpha+1}$ on the nodes introduced at this level of refinement and by not changing all the old basis functions. Obviously we have $V_1(\Omega) \subset V_2(\Omega) \subset \ldots \subset V_\alpha(\Omega) \subset V_{\alpha+1}(\Omega) \subset \ldots \subset V(\Omega)$ We have the relation $V_{\alpha+1} = V_\alpha \oplus \nu_{\alpha+1}$ where $\nu_{\alpha+1}$ is the subspace of $V_{\alpha+1}$ consisting of all finite element functions vanishing in the nodes of $\mathcal{T}_k$, of level l with $1 \leq l \leq k$. Therefore, that means the hierarchical basis of $V_{\alpha+1}$ is the direct sum of the hierarchical basis of V_α and the nodal basis of $\nu_{\alpha+1}$.

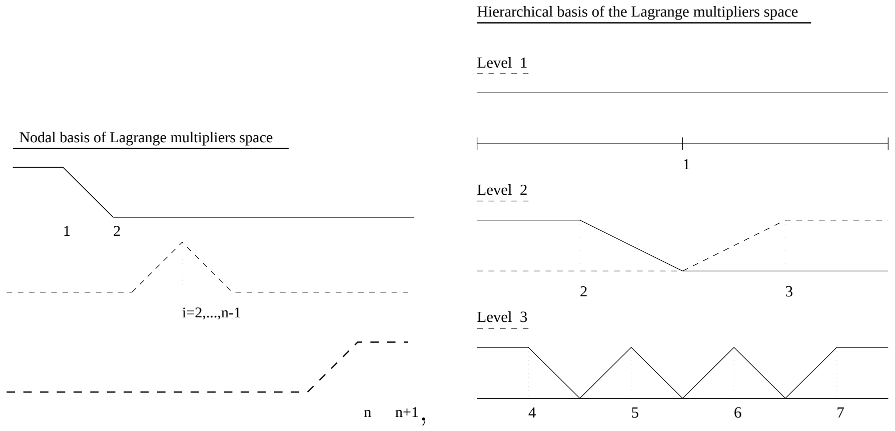
The next figure shows how we choose the Lagrange multipliers space.

4 A Block Diagonal Preconditioner

We remind that finally we arrive to an algebraic saddle point problem:

$$\begin{pmatrix} A & B^t \\ B & 0 \end{pmatrix} \begin{pmatrix} U \\ \lambda \end{pmatrix} = \begin{pmatrix} F \\ 0 \end{pmatrix} \tag{4.6}$$

Figure 1 The Lagrange multiplier space


Hierarchical basis of the Lagrange multipliers space
Level 1
1
Level 2
2
3
Level 3
4
5
6
7
Nodal basis of Lagrange multipliers space
1
2
i=2,...,n-1
n
n+1,

where A is a block diagonal matrix and B is the interface matrix, the jump operator. The system is equivalent to

$$\mathcal{A}\begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \tag{4.7}$$

Let $\mathcal{B}$ be a symmetric and positive definite matrix of the same size as $\mathcal{A}$.
Suppose that eigenvalues of the spectral problem $\mathcal{A}x = \nu\mathcal{B}x$ belong to the union of the segments $[d_1;d_2] \cup [d_3;d_4]$ where $d_1 \leq d_2 < 0 < d_3 \leq d_4$. Then it is possible to implement the generalized Lanczos method of minimal iterations to solve the saddle point problem $\mathcal{A}x = y$.
In [Kuz95], to give the motivation of their choice for the preconditioner $\mathcal{B}$, the eigenvalue problem is considered:

$$\begin{pmatrix} A & B^t \\ B & 0 \end{pmatrix}\begin{pmatrix} u \\ \lambda \end{pmatrix} = \nu \begin{pmatrix} A & 0 \\ 0 & BA^{-1}B^t \end{pmatrix}\begin{pmatrix} u \\ \lambda \end{pmatrix} \tag{4.8}$$

Obviously this problem can have only three nontrivial solutions $\{\frac{1-\sqrt{5}}{2}; 1; \frac{1+\sqrt{5}}{2}\}$.
A preconditioner $\mathcal{B}$ is taken with $R_u \sim A$ and $R_\lambda \sim BA^{-1}B^t$:

$$\mathcal{B} = \begin{pmatrix} R_u & 0 \\ 0 & R_\lambda \end{pmatrix} \tag{4.9}$$

Our idea is to test

$$R_u = A \quad \text{and } R_\lambda^{-1} = M = \sum (P_iP_i^t)^{-1}(P_iR_iA_iR_i^tP_i^t)(P_iP_i^t)^{-1}$$

5 Numerical Results

The Schur dual interface matrix $D = \sum_{i=1}^{K} B_iA_i^+B_i^t$ corresponds to the discretization of a compact operator. Because of this compactness, the eigenvalues of D accumulate towards 0 when h goes to 0, and the high end of the spectrum is less populated than the lower end. The spectral distribution of the interface problem has important consequences on the convergence rate of the conjugate gradient algorithm. During the first iterations, the conjugate gradient captures the eigenvalues corresponding to the low modes of the structure. Since D has only a few relatively high eigenvalues that correspond to the low physical modes of the structure, the CG algorithm applied to the solution of the dual problem gives quickly a good approximation of the displacement.

Figure 2 Spectral density with and without preconditioner, 2×2, $c = 1$, $h_1^{-1} = h_2^{-1} = 24, h_3^{-1} = h_4^{-1} = 32$

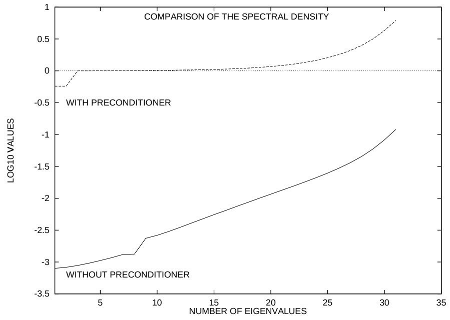


Figure 2 shows that the first preconditioner reduces the condition number of the cluster of small eigenvalues of the dual interface problem, and therefore, favors a superconvergence behavior of the CG algorithm. Figure 3 highlights the superconvergence effect.

Figure 3 Residual with hierarchical/nodal mortar space, 2×2, $c = 1, h_1^{-1} = h_2^{-1} = h_3^{-1} = h_4^{-1} = 33$

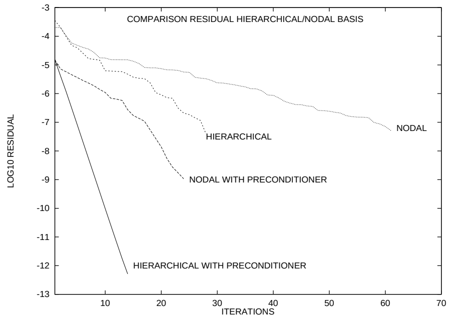
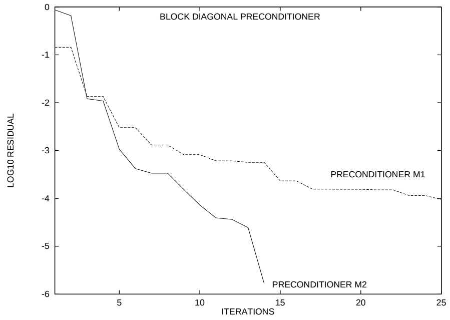


Figure 3 shows a comparison between the residual for the CG with and without preconditioner for the dual matrix written in its nodal and hierarchical basis. We have a very good convergence for the hierarchical matrix with the hierarchical preconditioner.

Figure 4 shows that the convergence is better with the preconditioner $R_\lambda^{-1} = M2 = \sum (P_i P_i^t)^{-1} (P_i R_i A_i R_i^t P_i^t)(P_i P_i^t)^{-1}$

Figure 4 Residual with $R_\lambda = M_1 = BA^{-1}B^t$ and $R_\lambda^{-1} = M_2 = \sum(P_iP_i^t)^{-1}(P_iR_iA_iR_i^tP_i^t)(P_iP_i^t)^{-1}$, 2 × 2, $c = 1$, $h_1^{-1} = h_2^{-1} = 16, h_3^{-1} = h_4^{-1} = 24$



REFERENCES

[BDY88] Bank R., Dupont T., and Yserentant H. (1988) The hierarchical basis multigrid method. *Numer. Math.* 52(31): 427+.

[BMP90] Bernardi C., Maday Y., and Patera A. (1990) A new nonconforming approach to domain decomposition: the mortar element method. *Publications du laboratoire d'Analyse Numérique de Paris VI* .

[Kuz95] Kuznetsov Y. (1995) Efficient iterative solvers for elliptic finite element problems on nonnatching grids. *J.Nuner.Anal.Math.Modelling* 10(3): 187+.

[RLJK96] Roux F., Lacour C., Japhet C., and Kalfon D. (March 1996) Méthodes de résolution par sous-domaines et calcul parallèle. Direction de l'Informatique RT 13/3717 CY, ONERA, 29, Av. de la Division Leclerc, Chatillon, FRANCE.

[Rou89] Roux F. (December 1989) *Méthode de décomposition de domaine à l'aide de multiplicateurs de Lagrange et application à la résolution en parallèle des équations de l'élasticité linéaire.* PhD dissertation, University Pierre et Marie Curie, 4, place Jussieu, Paris.

[Yse86] Yserentant H. (1986) On the multi-level splitting of finite element spaces. *Numer.Math.* 49(33): 379+.

49

Generalized Neumann-Neumann Preconditioners for Iterative Substructuring

Patrick Le Tallec and Marina Vidrascu

1 Introduction

In iterative substructuring, the parallel solution of a complex structural problem is achieved by splitting the original domain of computation in smaller nonoverlapping simpler subdomains, and by reducing the initial problem to an interface system with matrix

$$\mathbf{S} = \sum_i \mathbf{R}_i^t \mathbf{S}_i \mathbf{R}_i, \quad \mathbf{S}_i = \bar{\mathbf{K}}_i - \mathbf{B}_i^t (\mathring{\mathbf{K}}_i)^{-1} \mathbf{B}_i$$

to be solved by a parallel preconditioned conjugate gradient method. Many variants of this approach have been proposed and investigated in the recent literature, all associated to different choices of preconditioners. It turns out, in fact, that the interface problem requires specific preconditioners which take advantage of its particular structure. Such preconditioners must have nice parallel properties, must be able to handle arbitrary elliptic operators and discretization grids, and their performance must be insensitive to the discretization step h and to the number of subdomains. Many such preconditioners have appeared in the literature, following the early work of Bramble, Pasciak and Schatz [BPS86] ([CM94], [CMW93], [DW92], [Man90], [Wid88]). For three dimensional elasticity, efficient results have been obtained using either wire-basket algorithms such as proposed in Smith [Smi92] or Neumann-Neumann preconditioners ([DLV91], [LeT94]).

This last choice uses as preconditioner the following weighted sum of inverses [MB93] :

$$\mathbf{M}^{-1} = \mathbf{P} + (\mathbf{I} - \mathbf{P}) \left(\sum_i \mathbf{D}_i \mathbf{S}_i^{-1} \mathbf{D}_i^t \right) (\mathbf{I} - \mathbf{P}),$$

with $\mathbf{P}$ a coarse projection operator, and $\mathbf{D}_i$ a local partition of unity to be adapted to coefficients heterogeneities. This preconditioner is very general and can be applied to linear or nonlinear three dimensional elasticity problems using either matching or

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

non matching grids [LSV94], to nonlinear plates or shells problems [LMVed], or to incompressible flow problems [LP96].

It turns out that all these situations can be described and analyzed by a unique abstract framework. Indeed, the Neumann-Neummann algorithm is a standard additive Schwarz algorithm based on an interface space decomposition of the type

$$\mathbf{V} = \mathbf{V}_0 + \sum_i (\mathbf{I} - \mathbf{P})\mathbf{D}_i \mathbf{V}_i.$$

The purpose of this paper is to explain how to efficiently relate the Neumann-Neumann algorithm to the more classical additive Schwarz framework. The previously known convergence results of J. Mandel or of the authors are then easily recovered. More important, this framework leads to several extensions of the algorithm for situations involving inexact domain solvers or nonconforming 3D mesh refinements. The efficiency of these different extensions will be illustrated by the results of several real life numerical experiments.

2 Model Problem and Basic Algorithm

Let us consider a second order elliptic problem with vector unknown $u(x) \in \mathbb{R}^3$ set on a given domain Ω of $\mathbb{R}^3$ with variational formulation

$$\int_\Omega \Big(a(x) \cdot \nabla u(x)\Big) \cdot \nabla v(x) dx \; = \int_\Omega f^\Omega \cdot v dx + \int_{\partial\Omega_N} f^\Gamma \cdot v da, \quad \forall v \in H(\Omega). \tag{2.1}$$

Here, $H(\Omega)$ denotes the space of admissible (finite element) solutions, $\partial\Omega_N$ the part of the boundary where Neumann boundary conditions are imposed and $\partial\Omega_D$ the part where Dirichlet boundary conditions are imposed.

Iterative substructuring techniques use non overlapping domain partitions which split the original domain into small disjoint subdomains and reduce the original problem to an interface problem solved by an iterative conjugate gradient method.

The first step is thus to split the domain into small local non overlapping subdomains

$$\bar{\Omega} = \bigcup_{i=1}^{N} \bar{\Omega}_i,$$

with interfaces

$$\Gamma_i = \partial\Omega_i \backslash \partial\Omega, \tag{2.2}$$

$$\Gamma = \cup_i \Gamma_i. \tag{2.3}$$

The second step is to construct the interface problem. Let $\mathbf{K}_i$ denote the stiffness matrix of the subdomain Ω_i

$$(\mathbf{K}_i)_{lm} = \int_{\Omega_i} \Big(a(x) \cdot \nabla \phi_l(x)\Big) \cdot \nabla \phi_m(x) dx$$

and

$$(F_i)_l = \int_{\Omega_i} f^{\Omega}(x) \cdot \phi_l(x)dx + \int_{\partial\Omega_N \cap \partial\Omega_i} f^{\Gamma}(x) \cdot \phi_l(x)da$$

the corresponding right hand side. These matrices and right hand sides can obviously be computed independently on each subdomain. For each subdomain, the degrees of freedom are then decomposed into internal degrees of freedom $\mathring{X}_i$ associated to nodes which are strictly inside the subdomain Ω_i, or on the external boundary and interface degrees of freedom $\bar{X}_i$ associated to nodes lying on the interface between two or more neighboring subdomains. With this partition, the subdomain stiffness matrix and right hand side take the form

$$\mathbf{K}_i = \begin{bmatrix} \mathring{\mathbf{K}}_i & \mathbf{B}_i \\ \mathbf{B}_i^t & \bar{\mathbf{K}}_i \end{bmatrix}, \qquad F_i = \begin{bmatrix} \mathring{F}_i \\ \bar{F}_i \end{bmatrix}. \tag{2.4}$$

Let us finally denote by $\bar{X} = \bigcup_i \bar{X}_i$ the entire set of interface degrees of freedom, and by $\bar{X}_i = \mathbf{R}_i \bar{X}$ the restriction of $\bar{X}$ on the boundary of Ω_i. Under this notation, after addition of the local contributions of all subdomains to the global stiffness matrix and right hand side, the linear system describing the global equilibrium of the domain Ω takes the block structured form

$$\begin{bmatrix} \mathring{\mathbf{K}}_1 & 0 & \cdot & 0 & \mathbf{B}_1\mathbf{R}_1 \\ 0 & \mathring{\mathbf{K}}_2 & \cdot & 0 & \mathbf{B}_2\mathbf{R}_2 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & \mathring{\mathbf{K}}_N & \mathbf{B}_N\mathbf{R}_N \\ \mathbf{R}_1^t\mathbf{B}_1^t & \mathbf{R}_2^t\mathbf{B}_2^t & \cdot & \mathbf{R}_N^t\mathbf{B}_N^t & \sum_i \mathbf{R}_i^t\bar{\mathbf{K}}_i\mathbf{R}_i \end{bmatrix} \begin{pmatrix} \mathring{X}_1 \\ \mathring{X}_2 \\ \cdot \\ \mathring{X}_N \\ \bar{X} \end{pmatrix} = \begin{pmatrix} \mathring{F}_1 \\ \mathring{F}_2 \\ \cdot \\ \mathring{F}_N \\ \sum_i \mathbf{R}_i^t\bar{F}_i \end{pmatrix}.$$

This system is ideally solved by block Gaussian elimination of the internal degrees of freedom $\mathring{X}_i$, yielding

$$\mathring{X}_i = (\mathring{\mathbf{K}}_i)^{-1}(\mathring{F}_i - \mathbf{B}_i\bar{X}_i). \tag{2.5}$$

In mathematical terms, this elimination amounts to the parallel solution of local equilibrium problems set on subdomains Ω_i with fixed Dirichlet boundary conditions. After elimination we obtain the reduced interface system

$$\sum_i \mathbf{R}_i^t \left(\bar{\mathbf{K}}_i\mathbf{R}_i\bar{X} + \mathbf{B}_i^t(\mathring{\mathbf{K}}_i)^{-1}\left[\mathring{F}_i - \mathbf{B}_i\mathbf{R}_i\bar{X}\right]\right) = \sum_i \mathbf{R}_i^t\bar{F}_i.$$

Introducing the so-called local Schur complement matrix

$$\mathbf{S}_i := \bar{\mathbf{K}}_i - \mathbf{B}_i^t(\mathring{\mathbf{K}}_i)^{-1}\mathbf{B}_i, \tag{2.6}$$

this interface system takes the final form :

$$(\sum_i \mathbf{R}_i^t\mathbf{S}_i\mathbf{R}_i)\bar{X} = \sum_i \mathbf{R}_i^t(\bar{F}_i - \mathbf{B}_i^t(\mathring{\mathbf{K}}_i)^{-1}\mathring{F}_i). \tag{2.7}$$

Problem (2.7) is equivalent to our original equilibrium problem, but is only written in terms of the interface unknowns $\bar{X}$.

The main idea of modern domain decomposition methods is to solve problem (2.7) by an iterative preconditioned conjugate gradient algorithm. These iterative techniques never require the explicit calculation of matrix $\mathbf{S}$ since they form the matrix vector product $\mathbf{S}\bar{X}$ by solving auxiliary Dirichlet problems (2.5) on the local subdomains [LV96]. The main issue conditioning the success and parallel efficiency of such techniques is then the choice of the preconditioner $\mathbf{M}$. This preconditioner must be easy to implement in a parallel environment and must lead to a scalable algorithm when the number of processors increases. Additive Schwarz methods give a very general and efficient way of constructing such preconditioners.

3 Abstract Additive Schwarz Method

Let us consider the solution of the abstract variational problem

$$a(u,v) = \langle f, v\rangle, \forall v \in \mathbf{V}, u \in \mathbf{V}, \tag{3.8}$$

where $\mathbf{V}$ is a given Hilbert space with duality product $\langle .,. \rangle$, and a an elliptic continuous symmetric bilinear form defined on $\mathbf{V}$.

We suppose that the space $\mathbf{V}$ can be decomposed into the sum

$$\mathbf{V} = I_0\mathbf{V}_0 + I_1\mathbf{V}_1 + I_2\mathbf{V}_2 + \ldots + I_N\mathbf{V}_N, \tag{3.9}$$

where I_i is a given continuous linear extension map from the local space $\mathbf{V}_i$ to the global space $\mathbf{V}$. On each subspace $\mathbf{V}_i$, we introduce a symmetric elliptic bilinear form $b_i(.,.)$. We denote by $A : \mathbf{V} \to \mathbf{V}'$ the linear operator associated to the form a

$$\langle Au, v\rangle = a(u,v), \forall u, v \in \mathbf{V},$$

and by $B_i : \mathbf{V}_i \to \mathbf{V}_i'$ the linear operator associated to the form b_i.

With this notation, the additive Schwarz method for solving our original problem (3.8) is defined as the conjugate gradient method preconditioned by the following sum of local operators

$$M^{-1} = I_0 B_0^{-1} I_0^t + \ldots + I_N B_N^{-1} I_N^t. \tag{3.10}$$

This preconditioner is quite easy to compute since its action on a given element $L \in \mathbf{V}'$ is simply equal to the sum

$$M^{-1}L = \sum_{i=0}^{N} I_i u_i,$$

where $u_i \in \mathbf{V}_i$ is the solution of the local variational problem

$$b_i(u_i, v_i) = \langle L, I_i v_i\rangle, \forall v_i \in \mathbf{V}_i.$$

As usual, the efficiency of the above preconditioned conjugate gradient method is inversely proportional to the condition number of the operator $M^{-1}A$ which we control by carefully choosing the subspaces I_iV_i [CM94].

4 Generalized Neumann-Neumann Preconditioner

We have seen earlier that the interface problem (2.7) takes the abstract form

$$(\sum_i \mathbf{R}_i^t \mathbf{S}_i \mathbf{R}_i)\bar{X} = \bar{F} \in \mathbf{V}', \tag{4.11}$$

with $\mathbf{R}_i$ the restriction from the space $\mathbf{V}$ of global interface values $\bar{X}$ to the space $\mathbf{V}_i$ of local interface values $\bar{X}_i$. Such an abstract problem can be solved in all generality by a Neumann-Neumann algorithm which preconditions the sum $\mathbf{S} = \sum \mathbf{R}_i^t \mathbf{S}_i \mathbf{R}_i$ by a two level weighted sum of the inverses $\mathbf{M}^{-1} = \sum \mathbf{D}_i(\mathbf{S}_i)^{-1}\mathbf{D}_i^t$. From a theoretical point of view, this algorithm turns out to be a particular case of the above additive Schwarz method.

For constructing such an abstract Neumann-Neumann preconditioner, we first need to *choose*

1. a partition of unity $\mathbf{D}_i : \mathbf{V}_i \to \mathbf{V}$ satisfying

$$\sum_{i=1}^{N} \mathbf{D}_i \mathbf{R}_i = \mathbf{Id}|_{\mathbf{V}}.$$

For implementation reasons (flexibility and parallelism), the map $\mathbf{D}_i$ must be as local as possible. The generic choice consists in defining $\mathbf{D}_i$ on each interface degree of freedom $v(P_l)$ by : $\mathbf{D}_i v(P_l) = \frac{\rho_i}{\rho} v(P_k)$ if the l degree of freedom of $\mathbf{V}$ corresponds to the k degree of freedom of $\mathbf{V}_i$, and by $\mathbf{D}_i v(P_l) = 0$, if not. Here ρ_i is a local measure of the stiffness of subdomain Ω_i (for example an average Young modulus on Ω_i) and $\rho = \sum_{P_l \in \Omega_j} \rho_j$ is the sum of ρ_j on all subdomains Ω_j containing P_l.

2. an approximate local operator $\tilde{\mathbf{S}}_i$ such that

$$\tilde{\mathbf{S}} = \sum \mathbf{R}_\mathbf{i}^\mathbf{t} \tilde{\mathbf{S}}_i \mathbf{R}_i$$

is spectrally equivalent to $\mathbf{S}$: $\omega_- \langle \mathbf{S}v, v\rangle \le \langle \tilde{\mathbf{S}}v, v\rangle \le \omega_+ \langle \mathbf{S}v, v\rangle, \ \ \forall v \in \mathbf{V}$.
Up to now, Neumann-Neumann methods used the original Schur complement $\mathbf{S}_i$ as a local operator, but our most recent tests and analysis show that one can choose different local operators $\tilde{\mathbf{S}}_i$. In practice, one uses the local Schur complement of a simplified (unrefined, undeformed, homogenized..) problem. The calculation of its local inverse will then reduce to the solution of an approximate Neumann problem.

3. a $\tilde{\mathbf{S}}_i$ orthogonal decomposition of each local space $\mathbf{V}_i, i = 1, 2, \cdots, N$, into

$$\mathbf{V}_i = \mathbf{V}_i^0 \oplus \mathbf{Z}_i.$$

Above, the local coarse space $\mathbf{Z}_i$ contains all potential local singularities, that is functions v_i whose extensions $\mathbf{D}_i v_i$ are of very large (usually H dependent) energy. In particular, the space $\mathbf{Z}_i$ must be such that

$$Ker\tilde{\mathbf{S}}_i \subset \mathbf{Z}_i \subset \mathbf{V}_i.$$

For elasticity problems, $\mathbf{Z}_i$ is usually taken as the space of local rigid body motions. For plate and shell problems, a better choice is to choose [LMVed] $\mathbf{Z}_i$ as the orthogonal to the space

$$\mathbf{V}_i^0 = \{v \in \mathbf{V}_i, v = 0 \text{ at cross points}\}.$$

We then define the generalized Neumann-Neumann domain decomposition technique as the additive Schwarz algorithm solving $\mathbf{S}$ on the space $\mathbf{V}$ of interface restrictions of elements of $H(\Omega)$, with

1. coarse space $\mathbf{V}_0 = \sum_{i=1}^N \mathbf{D}_i\mathbf{Z}_i \subset \mathbf{V}$, endowed with the scalar product $\tilde{\mathbf{S}}$,
2. local spaces $\mathbf{V}_i^0, i = 1, 2, \cdots, N$ endowed with the scalar product $\mathbf{B_i} = \tilde{\mathbf{S}}_i$,
3. extensions $\mathbf{I}_i = (I - \mathbf{P})\mathbf{D}_i$, with $\mathbf{P}$ the $\tilde{\mathbf{S}}$ orthogonal projection of $\mathbf{V}$ onto $\mathbf{V}_0$. The above choice of extension map is in fact the key point of the Neumann-Neummann algorithm.

By construction, this Neumann-Neummann algorithm corresponds to the preconditioning operator

$$\mathbf{M}^{-1} = \tilde{\mathbf{S}}_0^{-1} + \sum_i (I - \mathbf{P})\mathbf{D}_i\tilde{\mathbf{S}}_i^{-1}\mathbf{D}_i^t(I - \mathbf{P})^t,$$

in which we recognize a direct generalization of the expression initially proposed in [MB93].

To see how this abstract algorithm can be numerically implemented, we detail below the application of the operator $\mathbf{M}^{-1}$ to a given element r of $\mathbf{V}'$. From the above construction, we first need to project the residual onto the coarse space by solving the coarse problem

$$\langle \tilde{\mathbf{S}}u_0, v_0\rangle = \langle r, v_0\rangle, \forall v_0 \in \mathbf{V}_0,$$

to compute the local contributions u_i by solving in parallel the local "Neumann" problems

$$\begin{aligned} u_i \in \mathbf{V}_i^0 \quad : \qquad \langle \tilde{\mathbf{S}}_i u_i, v_i\rangle &= \langle r, (I - \mathbf{P})\mathbf{D}_i v_i\rangle, \\ &= \langle r - \tilde{\mathbf{S}}u_0, \mathbf{D}_i v_i\rangle \quad \forall v_i \in \mathbf{V}_i^0, \end{aligned}$$

to project these local contributions onto the coarse space

$$\langle \tilde{\mathbf{S}}(\textstyle\sum_i \mathbf{D}_i z_i), v_0\rangle = \langle \textstyle\sum_i \mathbf{D}_i u_i, \tilde{\mathbf{S}}v_0\rangle, \qquad \forall v_0 \in \mathbf{V}_0, z_i \in \mathbf{Z}_i,$$

and to set

$$\mathbf{M}^{-1}r = u_0 + \sum_{i=1}^N \mathbf{D}_i(u_i - z_i).$$

Using the general theory, we then have

Theorem 1 *The above abstract Neumann-Neumann preconditioner satisfies*

$$Cond(\mathbf{M}^{-1}\mathbf{S}) = \frac{\lambda_{max}(\mathbf{M}^{-1}\mathbf{S})}{\lambda_{min}(\mathbf{M}^{-1}\mathbf{S})} \leq \frac{(Ne + 1)\omega_+}{\omega_-} \max_i \sup_{v_i \in \mathbf{V}_i^0} \frac{\|\mathbf{D}_i v_i\|_{\tilde{S}}^2}{\|v_i\|_{\tilde{S}_i}^2},$$

with Ne the maximum number of neighbors of a given subdomain.

Proof. The minimal eigenvalue $\lambda_{min}(\mathbf{M}^{-1}\mathbf{S})$ can be bounded from below by the well-known partition lemma classically used in the analysis of additive Schwarz methods. For this purpose, we split any $v \in \mathbf{V}$ into

$$v = \mathbf{P}v + (I - \mathbf{P})v = I_0 v_0 + v_\perp.$$

From the local decomposition of each local space $\mathbf{V}_i$, each local component $\mathbf{R}_i v_\perp$ can be decomposed into $\mathbf{R}_i v_\perp = v_i + z_i, v_i \in \mathbf{V}_i^0, z_i \in \mathbf{Z}_i$. By introducing our partition of unity $\mathbf{D}_i$, we then have

$$\begin{aligned} v_\perp &= (I - \mathbf{P}) \sum_i \mathbf{D}_i \mathbf{R}_i v_\perp \\ &= \sum_i (I - \mathbf{P}) \mathbf{D}_i (v_i + z_i) \\ &= \sum_i (I - \mathbf{P}) \mathbf{D}_i v_i, \\ &= \sum_i I_i v_i. \end{aligned}$$

By orthogonality of the local decomposition, we then verify

$$\begin{aligned} \sum_i b_i(v_i, v_i) &= \sum_i \langle \tilde{\mathbf{S}}_i v_i, v_i \rangle \leq \sum_i \langle \tilde{\mathbf{S}}_i (v_i + z_i), v_i + z_i \rangle \\ &= \sum_i \langle \tilde{\mathbf{S}}_i \mathbf{R}_i v_\perp, \mathbf{R}_i v_\perp \rangle = \langle \tilde{\mathbf{S}} v_\perp, v_\perp \rangle. \end{aligned}$$

By orthogonality again, we have

$$\begin{aligned} b_0(v_0, v_0) + \sum_i b_i(v_i, v_i) &\leq \langle \tilde{\mathbf{S}} v_0, v_0 \rangle + \langle \tilde{\mathbf{S}} v_\perp, v_\perp \rangle \\ &= \langle \tilde{\mathbf{S}} v, v \rangle \leq \omega_+ \langle \mathbf{S} v, v \rangle. \end{aligned}$$

Thus, the partition lemma holds with $C_0 = \omega_+$, which implies that λ_{min} is bounded from below by $\frac{1}{\omega_+}$.

On the other hand, the derivation of an optimal upper bound for $\lambda_{max}(\mathbf{M}^{-1}\mathbf{S})$ requires specific orthogonality arguments which do not easily fit into the classical theory of additive Schwarz methods. Indeed, using orthogonality and the contraction properties of the projection $(I - \mathbf{P})$, we have

$$\begin{aligned} \langle \tilde{\mathbf{S}} \mathbf{M}^{-1} \tilde{\mathbf{S}} \bar{X}, \mathbf{M}^{-1} \tilde{\mathbf{S}} \bar{X} \rangle &= \langle \tilde{\mathbf{S}} (u_0 + \sum_i (I - \mathbf{P}) \mathbf{D}_i u_i), u_0 + \sum_i (I - \mathbf{P}) \mathbf{D}_i u_i \rangle \\ &= \langle \tilde{\mathbf{S}} u_0, u_0 \rangle + \langle \tilde{\mathbf{S}} \sum_i (I - \mathbf{P}) \mathbf{D}_i u_i, \sum_i (I - \mathbf{P}) \mathbf{D}_i u_i \rangle \\ &\leq \langle \tilde{\mathbf{S}} u_0, u_0 \rangle + \langle \tilde{\mathbf{S}} \sum_i \mathbf{D}_i u_i, \sum_i \mathbf{D}_i u_i \rangle. \end{aligned}$$

By introducing the number of neighbors N_i of a given subdomain Ω_i

$$N_i = \text{Number of } j \neq i, \exists u_i \in \mathbf{V}_i, \exists u_j \in \mathbf{V}_j, \langle \tilde{\mathbf{S}}\mathbf{D}_i u_i, \mathbf{D}_j u_j \rangle \neq 0,$$

the continuity constant of $\mathbf{D}_i$

$$c_i = \sup_{v_i \in \mathbf{V}_i^0} \frac{\langle \tilde{\mathbf{S}}\mathbf{D}_i v_i, \mathbf{D}_i v_i \rangle}{\langle \tilde{\mathbf{S}}_{\mathbf{i}} v_i, v_i \rangle}$$

and using Cauchy Schwarz, we then deduce

$$\begin{aligned}
\langle \tilde{\mathbf{S}}\mathbf{M}^{-1}\tilde{\mathbf{S}}\tilde{X}, \mathbf{M}^{-1}\tilde{\mathbf{S}}\tilde{X} \rangle &\leq \langle \tilde{\mathbf{S}}u_0, u_0 \rangle + \max_i (N_i + 1) \sum_i \langle \tilde{\mathbf{S}}\mathbf{D}_i u_i, \mathbf{D}_i u_i \rangle \\
&\leq \langle \tilde{\mathbf{S}}u_0, u_0 \rangle + \max_i (N_i + 1)\; c_i \sum_i \langle \tilde{\mathbf{S}}_{\mathbf{i}} u_i, u_i \rangle \\
&\leq \langle \tilde{\mathbf{S}}\bar{X}, Pu_0 \rangle + \max_i (N_i + 1)\; c_i \sum_i \langle \tilde{\mathbf{S}}\bar{X}, I_i u_i \rangle \\
&\leq [\max_i (N_i + 1)\; c_i] \langle \tilde{\mathbf{S}}\bar{X}, \mathbf{M}^{-1}\tilde{\mathbf{S}}\bar{X} \rangle \\
&\leq [\max_i (N_i + 1)\; c_i] \langle \tilde{\mathbf{S}}\bar{X}, \bar{X} \rangle^{\frac{1}{2}} \langle \tilde{\mathbf{S}}\mathbf{M}^{-1}\tilde{\mathbf{S}}\bar{X}, \mathbf{M}^{-1}\tilde{\mathbf{S}}\bar{X} \rangle^{\frac{1}{2}}.
\end{aligned}$$

The final result follows then by standard estimates

$$\begin{aligned}
\lambda_{max}(\mathbf{M}^{-1}\mathbf{S}) &= \max_{\bar{X}} \frac{\langle \mathbf{S}\bar{X}, \bar{X} \rangle}{\langle \mathbf{M}\bar{X}, \bar{X} \rangle} = \max_{\bar{X}} \frac{\langle \mathbf{S}\bar{X}, \bar{X} \rangle}{\langle \tilde{\mathbf{S}}\bar{X}, \bar{X} \rangle} \, \frac{\langle \tilde{\mathbf{S}}\bar{X}, \bar{X} \rangle}{\langle \mathbf{M}\bar{X}, \bar{X} \rangle} \\
&\leq \frac{1}{\omega_-} \max_{\bar{X}} \frac{\langle \tilde{\mathbf{S}}\mathbf{M}^{-1}\tilde{\mathbf{S}}\bar{X}, \mathbf{M}^{-1}\tilde{\mathbf{S}}\bar{X} \rangle^{\frac{1}{2}}}{\langle \tilde{\mathbf{S}}\bar{X}, \bar{X} \rangle^{\frac{1}{2}}} \leq \frac{1}{\omega_-} \max_i (N_i + 1)\; c_i.
\end{aligned}$$

If we particularize this abstract convergence result to specific elasticity problems with specific choices of local spaces and coarse grid operators , we recover the following quite general convergence theorem [LeT94], [LMVed] :

Theorem 2 *Using the above Neumann-Neumann preconditioner in the framework of three-dimensional linear elasticity problems or of plate problems, the condition number of the operator* $\mathbf{M}^{-1}\mathbf{S}$ *is bounded by*

$$Cond(\mathbf{M}^{-1}\mathbf{S}) \leq \frac{C}{\alpha_i^2}[1 + ln \max_i \frac{H_i}{h_i}]^2, \tag{4.12}$$

the constant C *being independent of the subdomains diameters* H_i*, discretization steps* h_i*, aspect ratios* α_i *and averaged coefficients* ρ_i*.*

The above result guarantees the scalability of the proposed algorithm with respect to the number of subdomains (H_i independence), and its robustness with respect to strongly heterogeneous elasticity coefficients (ρ_i independence). This independence with respect to coefficient jumps is due to our specific choice of weighting factors

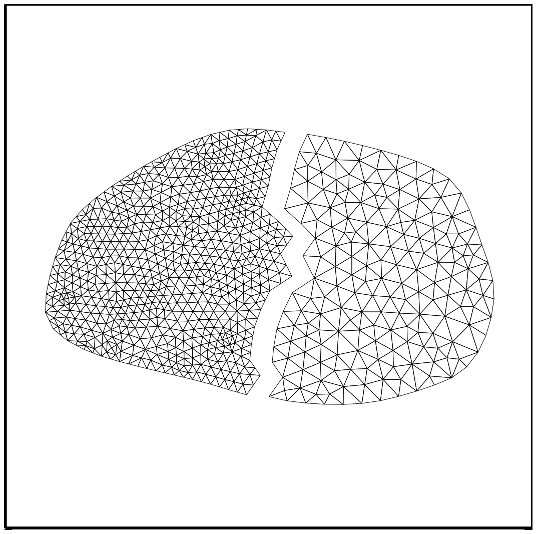
$$\mathbf{D}_i v(P_l) = \frac{\rho_i}{\rho} v(P_k).$$

But the abstract convergence result is more general because it handles situations using inexact subdomain solvers $\tilde{\mathbf{S}}_i^{-1}$ in the preconditioning step. The next paragraph illustrates this possibility.

5 Application to a Large-scale Problem with Local Refinement

Many large scale engineering problems require additional care and precision next to junctions of complex geometries. A simple way for achieving this in an industrial framework consists in first defining a global conforming finite element mesh of the whole domain Ω, to be partitioned as usual into conforming non overlapping subdomains Ω_i. In order to improve the local accuracy of the finite element solution, we then refine the finite element mesh of several subdomains (but not of all subdomains) by subdividing each original element of these subdomains into $2, 4, 8, 16, \ldots$ sub elements of same nature. After such local refinements the global mesh is no longer conforming : on the interface between a refined domain Ω_i and an unrefined domain Ω_j, several nodes of Ω_i will have no equivalent on Ω_j (Figure 1).

Figure 1 Nonconforming mesh refinement : refined nodes on the right domain have no counterpart on the left domain.



This lack of conformity can be handled by the so called slave node approach used in Bramble, Ewing, Parashkevov and Pasciak [BEPP92]. In this approach, all finite element displacement fields are imposed to be pointwise continuous at all subdomain interfaces. In other words, the finite element space definition is kept as

$$H(\Omega) \;=\; \Big\{ v_h : \bar{\Omega} \to \mathrm{I\!R}^3, v_h \text{ continuous}, \;\; v_h = 0 \text{ on } \partial\Omega_D,$$

$$v_{h_{|T_l}} = v_l \circ {'}_l^{-1}, v_l \in [Q_2'(\hat{\Omega})]^3, \;\text{for all elements } T_l \text{ of all subdomains } \Omega_i \Big\}.$$

With this choice, on any nonconforming interface, the values of the displacement field at any interface node P_l^i of the refined subdomain Ω_i which is not shared by the neighboring subdomain Ω_j are constrained to be equal to the value at this point of the Ω_j finite element interpolation of this field

$$u(P_l^i) = \sum_{k \in T \cap \partial\Omega_j} u(P_k^j)\phi_k^T(P_l^i).$$

Here T is the finite element of subdomain Ω_j which contains P_l^i, P_k^j are the interface nodes of this element, and $\phi_k^T(x)$ is the nodal element shape function associated to the node P_k^j. By construction, the interface nodes P_k^j of Ω_j are shared by Ω_i, and by the imposed continuity of the displacement field at the interface, the above continuity constraints can be rewritten as

$$u(P_l^i) = \sum_{k \in T \cap \partial\Omega_i} u(P_k^i)\phi_k^T(P_l^i). \tag{5.13}$$

Therefore, the additional degrees of freedom introduced by refinement on the interface $\partial\Omega_i$ are not directly related to any degree of freedom of Ω_j, but only to degrees of freedom of Ω_i. They must then be considered as internal degrees of freedom of Ω_i, that is as elements of the set $\overset{\circ}{X}_i$, and do not participate to the interface problem. In other words, the mesh refinement of the subdomain Ω_i will modify the local stiffness matrix $\mathbf{K}_i$ (new finite elements are added and the internal kinematic constraint (5.13) must be taken into account), will add elements to the set $\overset{\circ}{X}_i$ of internal unknowns, but will not modify the list and definition of the interface degrees of freedom $\bar{X}_i$. In particular, the interface problem keeps the same structure and dimension as in the unrefined case. Such situations can therefore be easily solved by our generalized Neumann-Neumann algorithm, using as interface preconditioner $\tilde{\mathbf{S}}$ the (much cheaper) interface preconditioner of the unrefined case.

We have applied this strategy to the calculation of part of a protection wall in a 3D offshore platform subjected to an external pressure. This problem can be written in the following form :

Find the displacement field $u(x)$, of a three-dimensional structure Ω, subjected to a given external loading. The external forces acting onto the body can be reduced to surface tractions f^Γ acting on the part $\partial\Omega_N$ of the boundary $\partial\Omega$. These tractions represent the external pressure or the action of icebergs on the structure. The displacement $u_i(x)$ is imposed on the remaining part $\partial\Omega_D = \partial\Omega - \partial\Omega_N$ of the boundary.

The governing equilibrium equations reduced then to the strong form :

$$\begin{aligned} -div(E(x)\,\varepsilon(u)) &= 0 \text{ in } \Omega, \\ E(x)\,\varepsilon(u)n &= f^{\Gamma} \text{ on } \partial\Omega_N, \\ u(x) &= 0 \text{ on } \partial\Omega_D, \end{aligned}$$

with E(x) the local elasticity tensor, and $\varepsilon(u)$ the linearized strain tensor

$$\varepsilon(u) = \frac{1}{2}\left(\nabla u + (\nabla u)^t\right).$$

For isotropic materials, we have simply

$$E(x)\,\varepsilon(u) = \frac{E\nu}{(1+\nu)(1-2\nu)} div(u) Id + \frac{E}{(1+\nu)}\varepsilon(u).$$

The platform is supposed to be made of an isotropic elastic material ($E = .3710^{11}$ and $\nu = .2$) and is discretized using second order hexahedral finite elements. The domain is cut into 5 subdomains. Three calculations were performed. The first one uses 5 coarse compatible subdomains. The total mesh is rather coarse, with element aspect ratios of 5. Four subdomains are of equal size and contain each 209 elements, 1616 nodes and 4848 degrees of freedom. The fifth subdomains has 88 elements, 714 nodes (2142 d.o.f). Then this fifth subdomain is refined (Figure 2) and thus the resulting decomposition uses non matching grids. The refined domain contains 748 elements, 4181 nodes (12543 d.o.f). Finally all the subdomains were refined. In order to fit in the computer memory a decomposition in 18 subdomains was needed for the fully refined case. The number of subdomain iterations, for a precision of 10^{-6} in the conjugate gradient algorithm, is 53 for the coarse decomposition, 57 for the incompatible one and 65 for the fine one. For the coarse decomposition the calculated condition number is 408.24, compared to 457.75 for the incompatible one. In both cases, the dimension of the interface problem is 2436. As far as accuracy is concerned, the non matching grid gives results comparable to those of the totally refined case. In summary, partial refinement is as accurate as full refinement but is associated to an interface problem whose complexity and cost is as cheap as the unrefined case when solved by our generalized Neumann - Neumann technique.

Similar costs are also observed when solving nonlinear elasticity problems with frozen interface preconditioner [LV96], which validates the generalized Neumann Neumann algorithm in real life numerical examples.

REFERENCES

[BEPP92] Bramble J., Ewing R., Parashkevov R., and Pasciak J. (1992) Domain decomposition methods for problems with partial refinement. *SIAM J. Sci. Stat. Comp.* 13: 397–410.

[BPS86] Bramble J., Pasciak J., and Schatz A. (1986) The construction of preconditioners for elliptic problems by substructuring. *Math. Comp.* 47: 103–134.

[CM94] Chan T. and Mathew T. (1994) Domain decomposition algorithms. *Acta Numerica* .

[CMW93] Cowsar L., Mandel J., and Wheeler M. (1993) Balancing domain decomposition for mixed finite elements. Technical Report 93-08, Rice University, Department of Mathematical Sciences, Rice University.

[DLV91] De Roeck Y.-H., Le Tallec P., and Vidrascu M. (1991) Domain decomposition methods for large linearly elliptic three dimensional problems. *J. Comp. Appl. Math.* 34: 93–117.

[DW92] Dryja M. and Widlund O. (1992) Additive Schwarz Methods for Elliptic Finite Element Problems in Three-Dimensions. In Chan T., Keyes D., Scroggs G. M. S., and Voigt R. (eds) *Proceedings of the fifth international symposium on Domain Decomposition Methods for Partial Differential Equations, Norfolk, May 1991*. SIAM, Philadelphia.

[Le 94] Le Tallec P. (1994) Domain decomposition methods in computational mechanics. In *Computational Mechanics Advances*, number 1, pages 121–220. North-Holland.

[LMVed] Le Tallec P., Mandel J., and Vidrascu M. (submitted) A Neumann-Neumann Domain Decomposition Algorithm for Solving Plate and Shell Problem. *SIAM J. Numer. Anal* .

[LP96] Le Tallec P. and Patra A. (1996) Non-overlaping Domain Decomposition Methods For Adaptive *hp* approximations of Stokes Problem with Discontinuous Pressure Fields. Technical report 96-33, TICAM, Univ of Texas.

[LSV94] Le Tallec P., Sassi T., and Vidrascu M. (1994) Three-dimensional Domain Decomposition Methods with Nonmatching Grids and Unstructured Coarse Solvers. In Keyes D. and Xu J. (eds) *Proceedings of the seventh international symposium on Domain Decomposition Methods for Partial Differential Equations, Penn State, October 93*, pages 133–139. AMS, Providence.

[LV96] Le Tallec P. and Vidrascu M. (1996) *Solving Large Scale Structural Problems on Parallel Computers using Domain Decomposition Technique*, chapter 3. J. Wiley, M. Papadrakakis edition.

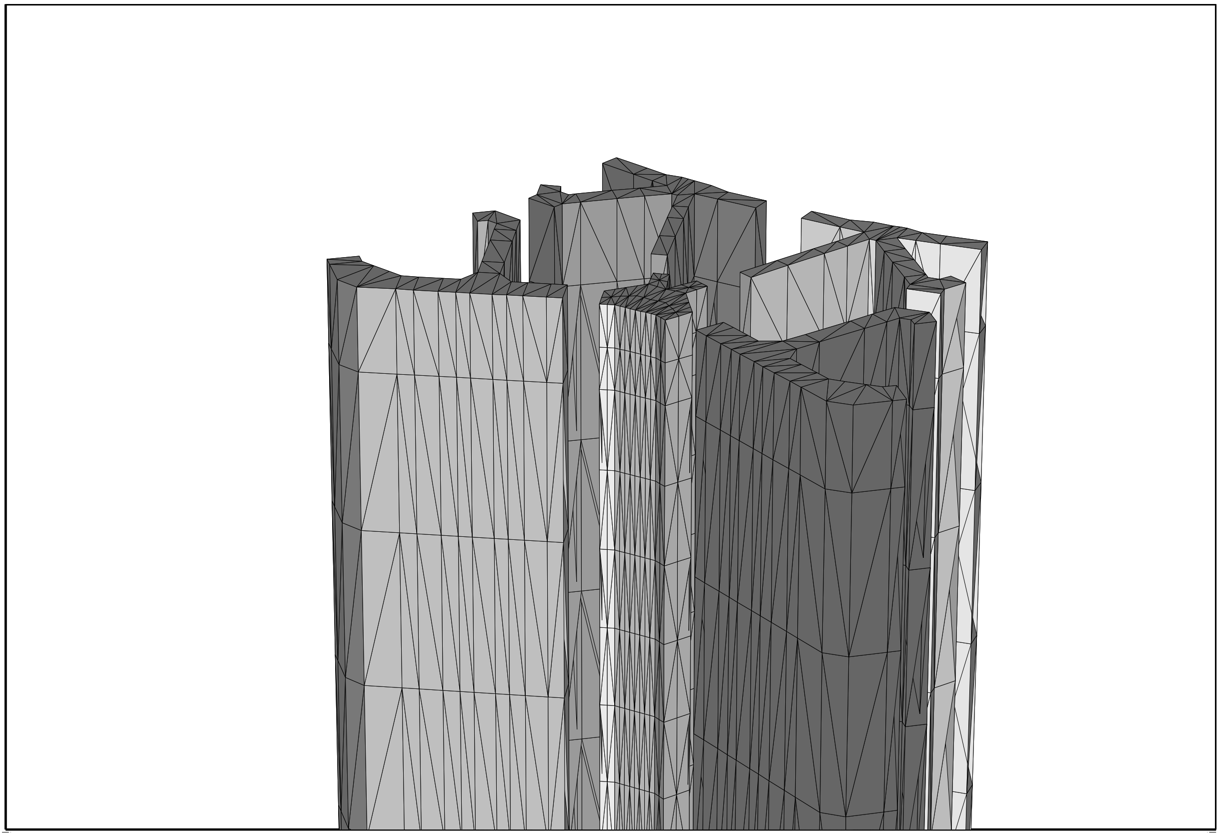
[Man90] Mandel J. (1990) Two-level domain decomposition preconditioning for the p-version finite element method in three dimensions. *Int. J. Num. Meth. Eng.* 29: 1095–1108.

[MB93] Mandel J. and Brezina M. (march 1993) Balancing Domain Decomposition: theory and performances in two and three dimensions. Technical report 7, Computational Mathematics Group, University of Colorado at Denver.

[Smi92] Smith B. (1992) An optimal domain decomposition preconditioner for the finite element solution of linear elasticity problems. *SIAM J. Sci Stat. Comp.* 13: 364–378.

[Wid88] Widlund O. (1988) Iterative substructuring methods: algorithms and theory for elliptic problems in the plane,. In Glowinski R., Golub G., and Periaux J. (eds) *Proceedings of the First International Symposium on Domain Decomposition Methods for Partial Differential Equations, Paris, France, January 7-9, 1987*. SIAM, Philadelphia.

Figure 2 Non matching grid decomposition of the offshore problem.



50

Some Recent Results on Domain Decomposition Methods for Eigenvalue Problems

S. H. Lui

1 Introduction

Domain decomposition methods for partial differential equations have traditionally been classified into two types: the Schwarz type where subdomains overlap and the Schur complement type where subdomains do not overlap. In this paper, we focus mainly on Schwarz algorithms for the eigenvalue problem for self–adjoint operators. Specifically, we study the model problem

$$-\triangle u = \lambda u \text{ on } \Omega$$

with homogeneous Dirichlet boundary conditions. Here, Ω is a bounded open domain with a smooth boundary. We shall discuss two different Schwarz algorithms with a brief mention of Schur algorithms. Some open problems will also be raised.

An incomplete list of papers on domain decomposition methods for the eigenvalue problems is [AGG88], [AG88], [Ben87], [Bou90], [Bou92], [Bd92], [Dri95] [DK92], [D'y96], [FG94], [HMV95], [Kny87], [KS89], [KS94], [Kro63], [Kuz86a], [Kuz86b], [LHL94], [Luo92], [Mal92], [Seh89], [Sim74] and [Sko91]. See also the references in these papers.

2 Schwarz Algorithms

Suppose the domain Ω is a union of $m > 1$ overlapping subdomains $\Omega_1 \cup \cdots \cup \Omega_m$. We discuss two Schwarz alternating methods.

Maliassov's Algorithm

This algorithm works with the variational formulation of the eigenvalue problem. Let (u, v) denote the usual $L^2(\Omega)$ inner product and $\|u\|^2 = (u, u)$. Denote the

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

energy inner product in the Sobolev space $H_0^1(\Omega)$ by $[u,v] = \int_\Omega \nabla u \cdot \nabla v$ and let $\|u\|_2 = \left(\int_\Omega (\triangle u)^2\right)^{1/2}$ for $u \in H^2(\Omega) \cap H_0^1(\Omega)$. Let the eigenvalues of $-\triangle$ be $\lambda_1 < \lambda_2 \leq \cdots$ and let ϕ_i be some eigenfunction corresponding to λ_i. For any $u \in H_0^1(\Omega) \setminus 0$, define the Rayleigh Quotient

$$R(u) = \frac{[u,u]}{(u,u)}.$$

Maliassov [Mal92] recently gave the following Schwarz Alternating Method to find the smallest eigenvalue and its associated eigenfunction and 'proved' its convergence.

Let $u^{(0)} \in H_0^1(\Omega) \setminus 0$ with $\lambda_1 \leq R(u^{(0)}) < \lambda_2$. For $n \geq 0$ and $1 \leq i \leq m$, define the sequence

$$\begin{aligned} \lambda^{(n+\frac{i}{m})} &= \inf\{R(u^{(n+\frac{i-1}{2})} + v_i);\ v_i \in H_0^1(\Omega) \setminus 0 \text{ with } v_i = 0 \text{ on } \Omega \setminus \Omega_i\} \\ &\equiv R(u^{(n+\frac{i}{m})}). \end{aligned}$$

(We identify the index $n + \frac{0}{m}$ with n and $n + \frac{m}{m}$ with $n+1$.) Then $\lim_{n\to\infty} \lambda^{(n)} = \lambda_1$ and a subsequence of $u^{(n)}$ converges to ϕ_1.

However, there is a small flaw in the algorithm. The fault lies in his definition of $u^{(n+\frac{i}{m})}$ which may not exist as the following example shows. Consider

$$-u'' = \lambda u \text{ on } (0, \tfrac{3\pi}{2})$$

with homogeneous Dirichlet boundary conditions. The smallest eigenvalue is $\frac{4}{9}$ with $\sin\frac{2x}{3}$ as a corresponding eigenfunction. Let Ω_1 be the interval $(0,\pi)$ and Ω_2 be the interval $(\frac{\pi}{2}, \frac{3\pi}{2})$. Take

$$u^{(0)} = \begin{cases} 2\sin x, & \text{if } 0 \leq x \leq \pi; \\ \sin 2x, & \text{if } \pi \leq x \leq \frac{3\pi}{2}. \end{cases}$$

Then $\lambda^{(0)} = \frac{4}{3}$. Let $D_1 = \{v_1 \in H_0^1(0, \frac{3\pi}{2}) \text{ with } v_1 = 0 \text{ on } [\pi, \frac{3\pi}{2}]\}$. Then,

$$\begin{aligned} \inf\{R(u^{(0)} + v_1);\ v_1 \in D_1\} &= \lim_{|a|\to\infty} R(u^{(0)} + aE\sin x) \\ &= 1, \end{aligned}$$

where $E\sin x$ is the function which is $\sin x$ on $[0,\pi]$ and is 0 on $[\pi, \frac{3\pi}{2}]$. This infimum cannot be attained by any $v_1 \in D_1$.

Despite this defect, Maliassov's main ideas are still valid. We now give a correct algorithm with a proof of convergence. Note that our result holds for an arbitrary eigenvalue, not just the smallest one. We restrict to the two-subdomain case.

Theorem 1 *Fix $p \in \mathbf{N}$. Define $H = \{f \in H_0^1(\Omega) \cap H^2(\Omega);\ (f,\phi_i) = 0,\ i = 1,\cdots,p-1\}$ and $D_i = \{f \in H;\ f = 0 \text{ on } \Omega \setminus \Omega_i\}$, $i = 1,2$. Let the initial guess be $u^{(0)} \in H \setminus 0$ with $\lambda^{(0)} = R(u^{(0)})$ smaller than the first eigenvalue strictly larger than λ_p. For $n \geq 0$ and $i = 1,2$, define the sequence*

$$\begin{aligned} \lambda^{(n+\frac{i}{2})} &= \inf\left\{R(cu^{(n+\frac{i-1}{2})} + v_i); c \in \mathbf{R}, v_i \in D_i, \|cu^{(n+\frac{i-1}{2})} + v_i\|_2 = 1\right\} \\ &\equiv R(u^{(n+\frac{i}{2})}). \end{aligned}$$

The above infimum is always attained. In case it is attained at more than one pair (c, v_i), any one can be taken to define $u^{(n+\frac{i}{2})}$. Then, $\lim_{n\to\infty} \lambda^{(n)} = \lambda_p$ and a subsequence of $u^{(n)}$ converges to ϕ_p in the energy norm.

Proof: We first show that the sequence is well–defined. Fix i and n. Let $c_j \in \mathbf{R}$ and $w_j \in D_i$ such that $z_j = c_j u^{(n+\frac{i-1}{2})} + w_j$ with $\|z_j\|_2 = 1$ and $R(z_j) \to \lambda^{(n+\frac{i}{2})}$ as $j \to \infty$. Since z_j is a bounded sequence in $(H, \|\cdot\|_2)$, there exists a subsequence, which we label by k_j, such that $z_{k_j} \to u^{(n+\frac{i}{2})}$ for some $u^{(n+\frac{i}{2})} \in H \setminus 0$ weakly in the norm $\|\cdot\|_2$ and strongly in the energy norm. Then $R(z_{k_j}) \to R(u^{(n+\frac{i}{2})})$ as $j \to \infty$ and the uniqueness of limit implies $\lambda^{(n+\frac{i}{2})} = R(u^{(n+\frac{i}{2})})$.

Since $\lambda^{(n+\frac{i}{2})}$ is a non–increasing sequence which is bounded below by λ_p, it must converge to some number, say, λ. Since $u^{(n)}$ is a bounded sequence in $(H, \|\cdot\|_2)$, there exists some subsequence which we label by n_j such that

$$u^{(n_j)} \to u$$

in the energy norm for some $u \in H \setminus 0$. Thus

$$\lim_{j\to\infty} R\left(u^{(n_j)}\right) = R(u) = \lambda.$$

We now show that (λ, u) is an eigenpair of $-\triangle$. Observe that for any $t \in \mathbf{R}$, $n \geq 1$ and $v_i \in D_i$, $i = 1, 2$,

$$R\left(u^{(n_j)} + tv_1\right) \geq \lambda^{(n_j+\frac{1}{2})}$$

and

$$R\left(u^{(n_j)} + tv_2\right) \geq R(u^{(n_j-1+\frac{1}{2})}) = \lambda^{(n_j-1+\frac{1}{2})}.$$

Taking the limit as $j \to \infty$ in the above inequalities, we obtain $R(u+tv_i) \geq \lambda,\ i = 1, 2$ which is equivalent to

$$\begin{aligned} t^2([v_i, v_i] - \lambda\|v_i\|^2) + 2t([u, v_i] - \lambda(u, v_i)) + [u, u] - \lambda\|u\|^2 &\geq 0 \\ t^2([v_i, v_i] - \lambda\|v_i\|^2) + 2t([u, v_i] - \lambda(u, v_i)) &\geq 0. \end{aligned}$$

This is possible only if $[u, v_i] = \lambda(u, v_i)$. Since the subdomains are overlapping, $H = D_1 + D_2$. Now any $v \in H_0^1(\Omega) \cap H^2(\Omega)$ can be represented as $v = v_1 + v_2 + \sum_{i=1}^{p-1} a_i \phi_i$ with $v_i \in D_i$ and $a_i \in \mathbf{R}$. Noting that $(u, \phi_l) = 0,\ l = 1, \cdots, p-1$, we obtain $[u, v] = \lambda(u, v)$. Since $H_0^1(\Omega) \cap H^2(\Omega)$ is dense in $H_0^1(\Omega)$, $[u, v] = \lambda(u, v)$ for all $v \in H_0^1(\Omega)$. Thus u is an eigenfunction with corresponding eigenvalue λ. By the variational principle for eigenvalues and the choice of initial guess, we must have $\lambda = \lambda_p$.

The general multiple–subdomain case is considered in [Lui96c]. ¿From the point of view of parallel computation, the above algorithm is not satisfactory because the computation on subdomain Ω_i must precede that on Ω_{i+1}. We now propose a version in which the computation in each subdomain can be carried out simultaneously. However, the calculation of the eigenvalue λ_p must precede that of λ_{p+1}. We shall consider the general m-subdomain case. The notation will be as in the previous theorem with the exception that we no longer identify an element indexed by $n+1$ with one indexed by $n + \frac{m}{m}$.

Theorem 2 *Fix $p \in \mathbf{N}$. Let the initial guess be $u^{(0)} \in H \setminus 0$ with $\lambda^{(0)} = R(u^{(0)})$ smaller than the first eigenvalue strictly larger than λ_p. For $n \geq 0$ and $1 \leq i \leq m$, define the sequences*

$$\begin{aligned} \lambda^{(n+\frac{i}{m})} &= \inf\{R(cu^{(n)} + v_i);\ c \in \mathbf{R}, v_i \in D_i, \|cu^{(n)} + v_i\|_2 = 1\} \\ &\equiv R(u^{(n+\frac{i}{m})}) \end{aligned}$$

and

$$\lambda^{(n+1)} = R(u^{(n+1)}) \equiv \min\left\{ R\left(\sum_{i=0}^{m} c_i u^{(n+\frac{i}{m})}\right);\ \sum_{i=0}^{m} c_i^2 = 1 \right\}.$$

The above infimum is always attained and in case the infimum is attained at more than one pair (c, v_i), then define $u^{(n+\frac{i}{m})}$ from any one of them. Similarly for the minimization problem. Then, $\lim_{n\to\infty} \lambda^{(n)} = \lambda_p$ and a subsequence of $u^{(n)}$ converges to ϕ_p in the energy norm.

The proof is very similar to that of the previous theorem and can be found in [Lui96c].

Note that in the statement of these theorems, only a subsequence converges. If the eigenvalue that we seek is a multiple eigenvalue, it is quite possible that different subsequences converge to different eigenfunctions corresponding to the same multiple eigenvalue. We are currently investigating whether the entire sequence converges in case the eigenvalue is simple.

If the initial Rayleigh Quotient is sufficiently large, then the Maliassov sequence for a 3-subdomain example converges to a different eigenvalue ([Lui96c]). These occurrences are rare and from our limited experience, the algorithms do converge globally in practice. We conjecture that global convergence holds for the 2-subdomain case. This article does not touch upon implementation issues. For an efficient hierarchical implementation and further theoretical results of Maliassov's algorithm, see the article by Chan and Sharapov elsewhere in this volume.

Another Schwarz Algorithm

We now discuss briefly another Schwarz algorithm which transmits information between the subdomains by boundary functions. Consider Ω to be an union of two overlapping subdomain Ω_1 and Ω_2. Let Γ_1 be $\partial\Omega_1 \cap \Omega_2$ and Γ_2 be $\partial\Omega_2 \cap \Omega_1$. The idea is to solve an eigenvalue problem in each subdomain with Robin boundary conditions. In this section, we are only concerned with the smallest eigenvalue and its associated eigenfunction which is nonzero in Ω.

For any positive integer i, define $u_2^{(i)}$ as the solution of the eigenvalue problem $-\triangle u_2^{(i)} = \lambda_2 u_2^{(i)}$ on Ω_2 with boundary conditions $u_2^{(i)} = 0$ on $\partial\Omega_2 \setminus \Gamma_2$ and $g_2^{(i)} u_2^{(i)} + \frac{\partial u_2^{(i)}}{\partial n_2} = 0$ on Γ_2, where $g_2^{(i)}$ is an estimate of the true boundary function on Γ_2 approximated by $u_1^{(i)}$:

$$g_2^{(i)} u_1^{(i)} + \frac{\partial u_1^{(i)}}{\partial n_2} = 0 \text{ on } \Gamma_2.$$

Here, n_i denotes the unit outward normal on Ω_i. Define $u_1^{(i+1)}$ as the solution of the eigenvalue problem on Ω_1 with boundary conditions $u_1^{(i+1)} = 0$ on $\partial\Omega_1 \setminus \Gamma_1$ and $g_1^{(i)} u_1^{(i+1)} + \frac{\partial u_1^{(i+1)}}{\partial n_1} = 0$ on Γ_1, where

$$g_1^{(i)} u_2^{(i)} + \frac{\partial u_2^{(i)}}{\partial n_1} = 0 \text{ on } \Gamma_1.$$

Note that $g_1^{(i)}$ is an estimate of the true boundary function on Γ_1 approximated by $u_2^{(i)}$. The above sequences are defined once we specify the initial iterate $u_1^{(1)}$. Note that we introduced the sequence of boundary functions $g^{(i)}$ for explanation purpose only. The actual boundary conditions can be simplified to

$$u_2^{(i)} \frac{\partial u_1^{(i+1)}}{\partial n_1} - \frac{\partial u_2^{(i)}}{\partial n_1} u_1^{(i+1)} = 0 \text{ on } \Gamma_1$$

and

$$u_1^{(i)} \frac{\partial u_2^{(i)}}{\partial n_2} - u_2^{(i)} \frac{\partial u_1^{(i)}}{\partial n_2} = 0 \text{ on } \Gamma_2.$$

We have not been able to show convergence of the above sequences. The method does converge in the few numerical experiments that we have tried. Local convergence and the exact rate of convergence for the one–dimensional problem is shown in [Lui96a]:

Theorem 3 *The Schwarz alternating method for the one–dimensional eigenvalue problem converges if the initial guess is sufficiently close to the true solution.*

For $0 < a < b < \pi$, let the subdomains covering $[0, \pi]$ be $(0, b)$ and (a, π). The sequences of eigenvalue problems reduce to

$$-u_1^{(i)''} = \lambda_1^{(i)} u_1^{(i)} \text{ on } (0, b), \;\; u_1^{(i)}(0) = 0, \;\; u_1^{(i)} u_2^{(i-1)'} - u_2^{(i-1)} u_1^{(i)'}\big|_{x=b} = 0$$

and

$$-u_2^{(i)''} = \lambda_2^{(i)} u_2^{(i)} \text{ on } (a, \pi), \;\; u_2^{(i)} u_1^{(i)'} - u_1^{(i)} u_2^{(i)'}\big|_{x=a} = 0, \;\; u_2^{(i)}(\pi) = 0$$

for $i = 1, 2, \ldots$. The sequences are defined after prescribing the 'initial condition' $u_2^{(0)}$. The exact solutions are:

$$u_1^{(i)}(x) = \sin(\alpha_i x), \quad u_2^{(i)}(x) = \sin(\beta_i(\pi - x)), \;\; i = 1, 2, \ldots$$

where the constants α_i and β_i are determined by the interior boundary conditions. After some algebra, we find that these constants are the smallest positive roots of the equations

$$\beta_{i-1} \cot\left(\beta_{i-1}(\pi - b)\right) + \alpha_i \cot(\alpha_i b) = 0 \tag{2.1}$$

and

$$\alpha_i \cot(\alpha_i a) + \beta_i \cot\left(\beta_i(\pi - a)\right) = 0, \;\; i = 1, 2, \ldots . \tag{2.2}$$

Once the value of β_0 has been specified, these sequences are well-defined. The proof of convergence reduces to showing that both the sequences α_i and β_i converge to one. The rate of convergence can be measured by

$$r_i = \left| \frac{\alpha_i - 1}{\alpha_{i-1} - 1} \right| \quad \text{and} \quad s_i = \left| \frac{\beta_i - 1}{\beta_{i-1} - 1} \right| .$$

It can be shown that

$$\lim_{i\to\infty} r_i = \lim_{i\to\infty} s_i = \frac{\cot(\pi - b) - \frac{\pi - b}{\sin^2(\pi - b)}}{\cot(\pi - a) - \frac{\pi - a}{\sin^2(\pi - a)}} \, \frac{\cot a - \frac{a}{\sin^2 a}}{\cot b - \frac{b}{\sin^2 b}} \equiv r.$$

For $0 < a < b < \pi$, it can be shown that $0 < r < 1$ and thus the sequences α_i and β_i converge to 1 at rate r asymptotically.

3 Schur Algorithms

The first domain decomposition algorithm for eigenvalue problems was derived by Kron [Kro63] and it is a Schur-type algorithm. Most of the papers listed earlier also belong to this category. For simplicity, let the domain consist of two non-overlapping subdomains Ω_1, Ω_2 with interface Γ separating them. Assume that the discrete eigenvalue problem can be written in the form

$$\begin{bmatrix} A_{11} & 0 & A_{13} \\ 0 & A_{22} & A_{23} \\ A_{13}^T & A_{23}^T & A_{33} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \lambda \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} . \tag{3.3}$$

Here, λ is some eigenvalue, u_i is the vector of unknowns in $\Omega_i, i = 1, 2$ and u_3 is the vector of unknowns along the interface. The matrices A_{ii} are assumed to be symmetric.

Formally, we may solve for u_1 and u_2 in terms of u_3. Substituting the results into the third equation in (3.3), we obtain

$$S(\lambda)u_3 \equiv \left[(A_{33} - \lambda) - A_{13}^T(A_{11} - \lambda)^{-1}A_{13} - A_{23}^T(A_{22} - \lambda)^{-1}A_{23}\right]u_3 = 0.$$

The matrix S is of dimension equal to the number of unknowns on the interface Γ and is thus much smaller than the size of the original matrix. Under some mild conditions ([Lui96b]), the eigenvalues of the original global matrix are precisely the values of λ at which $S(\lambda)$ has a zero eigenvalue. One way of accomplishing this is to find a root of the nonlinear equation $f(\lambda) \equiv \det S(\lambda) = 0$. It can be shown that f has poles at the union of the set of eigenvalues of A_{11} and of A_{22}. If an initial guess is not very close to the desired eigenvalue, it is quite possible that an iterate of Newton's or secant method may jump to a different interval bounded by different poles of f.

For instance, we consider the case of finding the first (smallest) eigenvalue λ_1 of the global matrix with an upper bound γ_1 which is a simple pole of f. For simplicity, assume that λ_1 is the only eigenvalue less than γ_1. Because γ_1 is a pole of f, a natural method is to find the zero of the de-singularized function $g(\lambda) = (\lambda - \gamma_1)f(\lambda)$ using Newton's, secant or Muller's method safeguarded by bisection. See also [AGG88].

Once a zero eigenvalue of $S(\lambda)$ has been found, one inverse iteration, for instance, may be used to determine its corresponding eigenvector u_3. The other components u_1 and u_2 of an eigenvector of the global matrix can subsequently be found from subdomain solves. See [Lui96b] for some theoretical results concerning the relationships among poles and zeroes of the eigenvalues of S and the eigenvalues of the original matrix. In [LG96], the smallest eigenvalue was computed using inverse iteration and employing preconditioned Krylov space methods. An open problem here is how to compute interior eigenvalues without the explicit formation of the matrix S. The explicit formation of S permits us to use direct methods to compute its inertia which in turn permits us to compute any specified eigenvalue ([Seh89]). Currently, there is no known fast method to determine the inertia of a matrix using only the knowledge of the action of the matrix on a vector. Without knowing the inertia, it does not seem possible to have an algorithm which guarantees that a prescribed eigenvalue is found.

Acknowledgement

I am grateful to Prof Gene Golub for introducing the topic of domain decomposition methods for eigenvalue problems to me. I thank Prof Tony Chan for a valuable discussion on Maliassov's algorithm leading to improved algorithms presented here.

REFERENCES

[AG88] Arbenz P. and Golub G. H. (1988) On the spectral decomposition of hermitian matrices modified by low rank perturbations with applications. *SIAM J. Matrix. Anal. Appl.* 9: 40–58.

[AGG88] Arbenz P., Gander W., and Golub G. H. (1988) Restricted rank modification of the symmetric eigenvalue problem: Theoretical considerations. *Linear Algebra and its Appl.* 104: 75–95.

[Bd92] Bourquin F. and d'Hennezel F. (1992) Numerical study of an intrinsic component mode sysnthesis method. *Computer Methods in Applied Mech. and Eng.* 97: 49–76.

[Ben87] Bennighof J. K. (1987) Component mode iteration for frequency calculations. *AIAA* 25(7): 996–1002.

[Bou90] Bourquin F. (1990) Analysis and comparison of several component mode synthesis methods on one–dimensional domains. *Numer. Math.* 58: 11–34.

[Bou92] Bourquin F. (1992) Component mode synthesis and eigenvalues of second order operators: Discretization and algorithm. *Mathematical Modelling and Numerical Analysis* 26: 385–423.

[DK92] D'Yakonov E. G. and Knyazev A. V. (1992) On an iterative method for finding lower eigenvalues. *Russ. J. Numer. Anal. Math. Modelling* 7(6): 473–486.

[Dri95] Driscoll T. A. (1995) Eigenmodes of isospectral drums. Technical report, Cornell University.

[D'y96] D'yakonov E. G. (1996) *Optimization in Solving Elliptic Problems.* CRC Press, Boca Raton.

[FG94] Farhat C. and Geradin M. (1994) On a component mode synthesis method and its application to incompatible substructures. *Computers & Structures* 51: 459–473.

[HMV95] Hitziger T., Mackens W., and Voss H. (1995) A condensation-projection method for the generalized eigenvalue problem. In Power H. and Brebbia C. A. (eds) *High Performance Computing in Engineering*, volume 1, pages 239–282.

Computational Mechanics Publications, Boston.

[Kny87] Knyazev A. V. (1987) Convergence rate estimates for iterative methods for a mesh symmetric eigenvalue problem. *Sov. J. Numer. Anal. Math. Modelling* 2: 371–396.

[Kro63] Kron G. (1963) *Diakoptics.* Macdonald, London.

[KS89] Knyazev A. V. and Skorokhodov A. L. (1989) Preconditioned iterative methods in subspace for solving linear systems with indefinite coefficient matrices and eigenvalue problems. *Sov. J. Numer. Anal. Math. Modelling* 4(4): 283–301.

[KS94] Knyazev A. V. and Skorokhodov A. L. (1994) Preconditioned gradient–type iterative methods in a subspace for partial generalized symmetric eigenvalue problems. *SIAM J. Num. Anal.* 31: 1225–1239.

[Kuz86a] Kuznetsov Y. A. (1986) Fictitious component and domain decomposition methods for the solution of eigenvalue problems. In Glowinski R. and Lions J. L. (eds) *Computing Methods in Applied sciences and Engineering VII*, pages 113–216. Elsevier Science Publishers, Amsterdam.

[Kuz86b] Kuznetsov Y. A. (1986) Iterative methods in subspaces for eigenvalue problems. In Balakrishnan A. V., Dorodnitsyn A. A., and Lions J. L. (eds) *Vistas in Applied Mathematics*, pages 96–113. Optimization Software, Inc., New York.

[LG96] Lui S. H. and Golub G. H. (1996) The use of preconditioning for the symmetric eigenvalue problem in domain decomposition. *preprint* .

[LHL94] Liew K. M., Hung K. C., and Lim M. K. (1994) On the use of the domain decomposition method for vibration of symmetric laminates having discontinuities at the same edge. *J. Sound and Vibration* 178(2): 243–264.

[Lui96a] Lui S. H. (1996) Domain decomposition methods for eigenvalue problems. *preprint* .

[Lui96b] Lui S. H. (1996) Kron's method for symmetric eigenvalue problems. *preprint* .

[Lui96c] Lui S. H. (1996) On two schwarz alternating methods for the symmetric eigenvalue problem. *preprint* .

[Luo92] Luo J. C. (1992) A domain decomposition method for eigenvalue problems. In Keyes D. E., Chan T. F., Meurant G. A., Scroggs J. S., and Voigt R. G. (eds) *Proc. Fifth Int. Conf. on Domain Decomposition Methods*, pages 306–321. SIAM, Philadelphia.

[Mal92] Maliassov S. Y. (1992) On the analog of Schwarz method for spectral problems. *Num. Meth. Math. Model.* pages 70–79 (in Russian).

[Seh89] Sehmi N. S. (1989) *Large Order Structural Eigenanalysis Techniques.* Ellis Horwood Ltd., New York.

[Sim74] Simpson A. (1974) Scanning Kron's determinant. *Quart. J. Mech. Appl. Math.* 27: 27–43.

[Sko91] Skorokhodov A. L. (1991) Domain decomposition method in partial symmetric eigenvalue problems. In Glowinski R., Kuznetsov Y. A., Meurant G. A., Periaux J., and Widlund O. B. (eds) *Proc. Fourth Int. Conf. on Domain Decomposition Methods for Partial Differential Equations*, pages 82–87. SIAM, Philadelphia.

51

Second-Order Transmission Conditions for the Helmholtz Equation

Jim Douglas, Jr. and Douglas B. Meade

1 Introduction

Several domain decomposition methods for the solution of elliptic problems have been proposed, analyzed, and successfully implemented during the past decade [BW86, BPS86, GW88, HTJ88, Lio88, Lio90]. In recent years these ideas have been extended to non-elliptic equations such as the Helmholtz equation [Des91, Des93, Des95, Ben95] and the harmonic Maxwell system [DJR92].

It is well-known [Lio88] that iterative methods using the Dirichlet or Neumann transmission conditions may not converge to the exact solution; the use of (first-order) Robin conditions on the inter-domain boundaries assures the convergence of the sequence of iterates. In practice, however, this convergence tends to be very slow. In this paper a set of second-order Robin-type transmission conditions is proposed. The new transmission conditions significantly improve the rate of convergence of the iterates. After a brief overview of the general problem, including a general domain decomposition formulation, the specific model with second-order Robin-type transmission conditions is presented. A numerical example is used to demonstrate the benefits of this method. The discussion in this paper is intended to motivate the new algorithm and illustrate the type of improvements that are possible. A full description of the method with the accompanying theory is under development.

2 General Problem

Many applications in electromagnetics, acoustics, and elasticity require the solution of a wave equation on an unbounded domain. A number of methods have been created for the reduction of the problem to a bounded domain. A common approach is to truncate the exterior domain and impose an appropriate boundary condition on the artificial boundary. The exact "radiation" boundary condition (RBC) is non-local (in both space

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

and time); numerous spatially local approximate RBCs have also been developed (see, *e.g.*, [Giv91]). Interest in domain decomposition methods for the solution of these problems arises from the fact that the direct solution of realistic scattering problems require the solution of large, sparse, complex-valued systems of linear equations. Domain decomposition methods are employed to create an iterative method requiring the direct solution of related problems on a small subdomain, typically a single biquadratic finite element.

The model problem selected for this investigation is the time-reduced scalar wave equation, *i.e.*, the Helmholtz equation, in the exterior, Ω^+, of a two-dimensional scatterer, Ω:

$$\begin{aligned} -\triangle u - \omega^2 u &= f && \text{in } \Omega^+ && (2.1)\\ u &= g_0 && \text{on } \partial\Omega && (2.2)\\ \left|\tfrac{\partial u}{\partial r} - i\omega u\right| &= o(r^{-1/2}) && \text{as } r \to \infty, \text{ uniformly in } \theta. && (2.3) \end{aligned}$$

Note that the Sommerfeld radiation condition, (2.3), prevents the creation of energy at infinity. Thus, the problem has at most one solution.

The corresponding problem on a bounded domain is obtained by truncating the domain at an artificial boundary, Γ^t, and replacing (2.3) with a RBC with tangential boundary operator, $\mathcal{B}$. That is,

$$\begin{aligned} -\triangle u - \omega^2 u &= f && \text{in } \Omega^t && (2.4)\\ u &= g_0 && \text{on } \Gamma && (2.5)\\ \tfrac{\partial u}{\partial n} + \mathcal{B}u &= g_j && \text{on } \Gamma^t. && (2.6) \end{aligned}$$

Selection of Γ^t and $\mathcal{B}$ should be made so that a solution to (2.4)–(2.6) is both a good approximation to the solution to (2.1)–(2.3) and can be numerically computed in an efficient manner. Balancing these opposing constraints can be difficult and, in practice, often depends on the specific application. For example, a particularly effective combination used in many electromagnetics problems is to place Γ^t about one wavelength from the (convex hull of the) scatterer and to use the Kriegsmann RBC for $\mathcal{B}$ [KTU89, LWMP96]. For a long, thin rectangular scatterer and a reasonable discretization of the resulting domain the linear system involves more than 7,000 unknowns. While this is a considerable savings over the system of more than 35,000 unknowns that results from the use of a circular artificial boundary, the benefits are seen in scattering problems. (The 3-D vector problem presents even more problems.)

3 Domain Decomposition Methods for the Helmholtz Equation

A nonoverlapping domain decomposition method is a natural choice for the iterative solution of (2.4)–(2.6) . Let Ω^t be partitioned into a finite number of nonoverlapping subdomains Ω_j. The interfaces between subdomains are denoted by Σ_{jk}; Γ_j and Γ_j^t denote the intersections of a subdomain with the scatterer and artificial boundary, respectively. That is, $\Omega^t = \bigcup_{j\in J} \Omega_j$, $\Sigma_{jk} := \partial\Omega_j \cap \partial\Omega_k$ for all $j \neq k$, $\Gamma_j := \partial\Omega_j \cap \Gamma$, and $\Gamma_j^t := \partial\Omega_j \cap \Gamma^t$. The outward unit normal vector, relative to Ω_j, is ν_j and $g_j := g|_{\Gamma_j^t}$.

Table 1 Lowest-order radiation boundary conditions, $\mathcal{B}u := \alpha u + \beta\frac{\partial^2 u}{\partial \tau^2}$.

Order	Type	α	β
0	Neumann	0	0
1	Robin	$i\omega$	0
2	Robin	$i\omega$	$\dfrac{i}{2\omega}$

The iterative domain decomposition algorithm requires an initial solution, u_j^0, often zero, on each subdomain, then computes the sequence u_j^n of functions that satisfies

$$\begin{aligned}
(-\Delta - \omega^2)u_j^{n+1} &= f && \text{in } \Omega_j && (3.7)\\
u_j^{n+1} &= g_0 && \text{on } \Gamma_j && (3.8)\\
(\tfrac{\partial}{\partial \nu_j} + \mathcal{B})u_j^{n+1} &= g_j && \text{on } \Gamma_j^t && (3.9)\\
(\tfrac{\partial}{\partial \nu_j} + \mathcal{T})u_j^{n+1} &= (-\tfrac{\partial}{\partial \nu_k} + \mathcal{T})u_k^n && \text{on } \Sigma_{jk} \quad \forall k && (3.10)
\end{aligned}$$

where $\mathcal{T}$ is the tangential differential operator used as the interface condition between adjacent subdomains.

The convergence of this method depends primarily on the choice of the tangential boundary operators $\mathcal{B}$ and $\mathcal{T}$. The Neumann and two lowest-order Robin-type radiation boundary operators are summarized in Table 1. It is well-known [Lio88, Lio90] that the Dirichlet and Neumann transmission conditions do not guarantee convergence of the iterations for all values of the frequency ω. Després [Des91, Des93, Des95] has shown that a convergent iterative method does result from the use of the first-order Robin-type boundary condition for both $\mathcal{B}$ and $\mathcal{T}$. The convergence is in both $H^1(\Omega_j)$ for all j and, under additional smoothness assumptions on the subdomains, in $H^{-\frac{1}{2}-\epsilon}(\Omega^t)$ for all $\epsilon \in (0, \frac{1}{2}]$. [1]

In practice, however, this algorithm exhibits a surprisingly slow rate of convergence [Des93]. A noticeable improvement in the rate of convergence is obtained if an under-relaxed version of the transmission condition is used, *i.e.*, replace (3.10) with

$$(\frac{\partial}{\partial \nu_j} + \mathcal{T})u_j^{n+1} = (1-\delta)(-\frac{\partial}{\partial \nu_k} + \mathcal{T})u_k^n + \delta(\frac{\partial}{\partial \nu_j} + \mathcal{T})u_j^n \quad \text{on } \Sigma_{jk} \quad \forall k \tag{3.11}$$

for some value of the relaxation parameter $\delta \in [0, 1)$.

A better approximation to the original wave propagation problem on an unbounded domain (2.1)–(2.3) is obtained when the second-order RBC is applied on the artificial boundary. It is conjectured that the use of the second-order transmission condition similarly improves the convergence of the domain decomposition method. The analysis

[1] While Després' results are developed for the special case in which there is no scatterer, *i.e.*, $\Gamma = \emptyset$, it is easily seen that the same holds for the more general problem.

of this problem is not substantially different from the analysis of the problem with first-order radiation and transmission conditions. The second-order tangential derivative introduces some additional technicalities into the analysis of this algorithm, but the same general approach can still be applied.

4 Variational Formulation

The simplicity of this method and its similarity to the first-order algorithm (and others of the same type) is clearly demonstrated by the variational formulation of the problem. Introduce the flux on the boundary and each interface as a Lagrange multiplier $\lambda_j := \left.\frac{\partial u}{\partial \nu_j}\right|_{\partial\Omega_j}$ (see, *e.g.*, [Des93]). Let the standard L_2 inner product be denoted by $(\cdot,\cdot)$ and the $H^{-1/2}$ — $H^{1/2}$ duality pairing by $\langle\cdot,\cdot\rangle$. The function space $\mathcal{H}(\Omega_j)$ contains all functions in $H^1(\Omega_j)$ with sufficient (tangential) smoothness on the boundary to assure that

$$\mathcal{B} : \mathcal{H}(\Omega_j) \to H^{-1/2}(\Gamma_j^t) \qquad \text{and} \qquad \mathcal{T} : \mathcal{H}(\Omega_j) \to H^{-1/2}(\Sigma_{jk}) \text{ for all } k.$$

The variational problem corresponding to the under-relaxed version of (3.7)–(3.10) is:

> given initial functions u_j^0 on Ω_j and λ_j^0 on $\partial\Omega_j$, find (for all j) the (complex-valued) functions $u_j^{n+1} \in \mathcal{H}(\Omega_j)$ with $u_j^{n+1} = g_0$ on Γ_j and $\lambda_j^{n+1} \in H^{-1/2}(\partial\Omega_j)$ such that
>
> $$\begin{aligned} \left(\nabla u_j^{n+1}, \nabla v\right)_{\Omega_j} &- \omega^2\left(u_j^{n+1}, v\right)_{\Omega_j} + \left\langle \mathcal{B} u_j^{n+1}, v\right\rangle_{\Gamma_j^t} + \sum_k \left\langle \mathcal{T} u_j^{n+1}, v\right\rangle_{\Sigma_{jk}} \\ &= \sum_k \left\langle \delta\lambda_j^n - (1-\delta)\lambda_k^n, v\right\rangle_{\Sigma_{jk}} \\ &\quad + \sum_k \left\langle \mathcal{T}\left(\delta u_j^n + (1-\delta)u_k^n\right), v\right\rangle_{\Sigma_{jk}} \\ &\quad + (f,v)_{\Omega_j} + \langle g_j, v\rangle_{\Gamma_j^t} \qquad (4.12) \\ \left\langle \lambda_j^{n+1}, w\right\rangle_{\Gamma_j^t} &= \langle g_j, w\rangle_{\Gamma_j^t} - \left\langle \mathcal{B} u_j^{n+1}, w\right\rangle_{\Gamma_j^t} \qquad (4.13) \\ \left\langle \lambda_j^{n+1}, w\right\rangle_{\Sigma_{jk}} &= -\left\langle \lambda_k^n, w\right\rangle_{\Sigma_{jk}} + \left\langle \mathcal{T}(u_k^n - u_j^{n+1}), w\right\rangle_{\Sigma_{jk}} \qquad (4.14) \end{aligned}$$
>
> for all (real-valued) test functions $v \in \mathcal{H}(\Omega_j)$ that vanish on Γ_j and $w \in H^{1/2}(\partial\Omega_j)$.

Note that, except under special conditions on $\mathcal{B}$ and the smoothness of the domains, these variational problems are not Hermitian.

5 Computational Results

To illustrate the improvements that can be expected from this algorithm, consider the following test problem. Let $\Omega^t := (0,1) \times (0,1)$ and $\Gamma := \emptyset$. Subdivide Ω^t into n

vertical strips, *i.e.*, for $j = 1, 2, \ldots, n$, $\Omega_j = (\frac{j-1}{n}, \frac{j}{n}) \times (0,1)$, $\Sigma_{jk} = \emptyset$ for $k \neq j+1$, and $\Sigma_{j,j+1} = \{(\frac{j}{n}, y) : y \in (0,1)\}$ $(j = 1, 2, \ldots, n-1)$. Let $\mathcal{B}u := \alpha u + \beta \frac{\partial^2 u}{\partial \tau^2}$ with coefficients taken from Table 1. On each subdomain the test and trial functions are chosen to be bi-quadratic.[2] Initialize both the solution, u_j^0, and Lagrange multipliers, λ_j^0, to zero. Computing the next iterate on one subdomain involves the solution of a 9×9 complex-valued linear system to compute u_j^{n+1} and four 3×3 complex-valued linear systems to compute λ_j^{n+1} along each edge.

All that remains is to select the data for the problem: f and g_j. Let U be a bi-quadratic function on Ω^t and choose $f = -\triangle U - \omega^2 U$ and $g_j = -\frac{\partial U}{\partial \nu_j} + \mathcal{B}U$, for each $j = 1, 2, \ldots, n$. Thus, the exact solution to (4.12)–(4.14) is $u = U$. The iterations terminate when the relative error of the solution and Lagrange multiplier on each subdomain, measured in the appropriate L_2-norm, falls below a specified threshold. That is, for a given $\epsilon > 0$, $\max_j \left\{ \left\| u_j^{n+1} - U \right\|_{L_2(\Omega_j)}, \left\| \lambda_j^{n+1} - \frac{\partial U}{\partial \nu_j} \right\|_{L_2(\partial\Omega_j)} \right\} < \epsilon$.A convergence test based on relative error might seem more appropriate, but some of the exact values of the Lagrange multipliers vanish in the examples of interest. In fact, since all relevant norms of the exact solution either vanish or exceed unity, the above absolute error test is actually a slightly more stringent condition.

Note that while this choice of data avoids all issues relating to approximation error, it is not consistent with the original scattering problem — U does not satisfy the Sommerfeld radiation condition. Regardless, this is still a valid test of a solution algorithm for the solution of the boundary value problem (2.4)–(2.6).

The optimal choice of the relaxation parameter is not known. The random selection of $\delta \in [0.7, 1)$ for each iteration is reported, by Després [Des93], to yield unexpectedly good results. In an effort to work with a deterministic algorithm for this project, a single value for δ must be selected; the value $\delta = 0.8$ appears to be close to optimal for a wide range of problems.

The results in Table 4, obtained using $\epsilon = 10^{-3}$, are representative of the performance that can be expected from this algorithm. In each case, the problems utilizing second-order radiation and transmission conditions converge faster than the corresponding problem with first-order conditions; the specific improvement ranges from 10% to 70% and averages a little more than 50%. The benefits of under-relaxation are also evident in all test cases. It is interesting, however, to note that the improvement due to under-relaxation is noticeably greater for the first-order problems.

6 Additional Issues

The experimental results are encouraging, but several issues remain unanswered. Partial answers are summarized where possible. Computational evidence referred to in this section is based on examples closely related to those presented here.

- Note that under-relaxation can be used, independently, on each term in the transmission condition. Is there any advantage to relaxing the two terms

[2] Note that while solutions to this problem are complex-valued, it suffices to use real-valued bases for the test and trial spaces.

Table 2 Comparison of iterations to convergence for first- and second-order Robin transmission conditions with and without under-relaxation.

		# Iterations to Convergence			
Exact		order 1		order 2	
Solution	Grid	$\delta = 0$	$\delta = 0.8$	$\delta = 0$	$\delta = 0.8$
1	2×1	214	13	63	12
1	4×1	285	27	100	22
1	8×1	517	69	177	40
1+x	2×1	239	153	122	58
1+x	4×1	415	258	212	122
(1+x)(1+y)	2×1	256	165	76	63
(1+x)(1+y)	4×1	445	277	151	130

(λ_j^{n+1} and $\mathcal{T}u_j^{n+1}$) by different amounts? Likewise, would other aspects of the problem benefit from the use of under-relaxation, smoothing, or other modification to the standard iteration?

- The results in [Des93] are based on the relative L_2-error of the solution; there is no guarantee that the Lagrange multipliers have converged. In fact, computational tests indicate that the Lagrange multipliers converge much slower than the solution in each subdomain.
- Choosing the initial solution to be zero is easy to implement. It is also somewhat simpler to analyze. Is there a better choice for the initial solution?
- Table 2 appears to indicate that the number of iterations is roughly linear in the number of vertical strips. While this general trend is observed in larger tests, the correlation seems to not be as strong as the results presented in Table 2 might suggest. This implies that the current implementation, with one element per subdomain, is not likely to be optimal for large problems. Is it possible to find an optimal balance between the selection of a decomposition, the efficiency of the subdomain solver, and the transmission of information between subdomains?
- These tests always require that $\mathcal{B} = \mathcal{T}$. Preliminary computational tests in which $\mathcal{B}$ and $\mathcal{T}$ are Robin-type boundary operators of different orders can, when combined with the appropriate use of under-relaxation, be convergent. More specifically, while the local Robin-type boundary operators are optimal (in a certain sense) for use on the truncation boundary, are the same operators the optimal choice for the transmission condition? Related work in this direction (see, *e.g.*, [HTJ88, GCJ95]) recommends the use of non-local transmission conditions.

REFERENCES

[Ben95] Benamou J.-D. (1995) A domain decomposition method for the optimal control of systems governed by the Helmholtz equation. In Bécache E., Joly P., and Roberts J. E. (eds) *Mathematical and Numerical Aspects of Wave Propagation Phenomena*, pages 653–662. SIAM.

[BPS86] Bramble J. H., Pasciak J. E., and Schatz A. H. (1986) The construction of preconditioners for elliptic problems by substructuring. *Math. Comp.* 46: 361–389.

[BW86] Björstad P. E. and Widlund O. B. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Num. Anal.* 23: 1097–1120.

[Des91] Després B. (1991) Domain decomposition method and the Helmholtz problem. In Cohen G., Halpern L., and Joly P. (eds) *Proc. First Int. Conf. on Mathematical and Numerical Aspects of Wave Propagation Phenomena*, pages 44–52. SIAM.

[Des93] Després B. (1993) Domain decomposition method and the Helmholtz problem (part II). In Kleinman R., Angell T., Colton D., Santosa F., and Stakgold I. (eds) *Proc. Second Int. Conf. on Mathematical and Numerical Aspects of Wave Propagation Phenomena*, pages 197–206. SIAM.

[Des95] Després B. (1995) An non-overlapping iterative linear domain decomposition method for the Helmholtz problem. preprint.

[DJR92] Després B., Joly P., and Roberts J. E. (1992) A domain decomposition method for the harmonic Maxwell's equations. In *Proc. of IMACS Int. Symposium on Iterative Methods in Linear Algebra*, pages 475–484. North Holland.

[GCJ95] Ghanemi S., Collino F., and Joly P. (1995) Domain decomposition method for harmonic wave equations. In Bécache E., Joly P., and Roberts J. E. (eds) *Mathematical and Numerical Aspects of Wave Propagation Phenomena*, pages 663–672. SIAM.

[Giv91] Givoli D. (1991) Non-reflecting boundary conditions. *J. Comp. Phys.* 94: 1–29.

[GW88] Glowinsky R. and Wheeler M. F. (1988) Domain decomposition and mixed finite element methods for elliptic problems. In Glowinski R., Golub G., Meurant G., and Périaux J. (eds) *Proc. First Int. Syposium on Domain Decomposition Methods for Partial Differential Equations*, pages 144–171. SIAM.

[HTJ88] Hagstrom T., Tewarson R. P., and Jazcilevich A. (1988) Numerical experiments on a domain decomposition algorithm for nonlinear elliptic boundary value problems. *Appl. Math. Lett.* 1: 299–302.

[KTU89] Kriegsmann G. A., Taflove A., and Umashankar K. R. (1989) A new formulation of electromagnetic wave scattering using an on-surface radiation condition approach. *IEEE Trans. Antennas Propagat.* 35: 153–161.

[Lio88] Lions P. L. (1988) On the Schwartz alternating method I. In Glowinski R., Golub G., Meurant G., and Périaux J. (eds) *Proc. of First Int. Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 1–42. SIAM.

[Lio90] Lions P. L. (1990) On the Schwartz alternating method III: A variant for nonoverlapping subdomains. In Chan T. F., Glowinski R., Périaux J., and Widlund O. (eds) *Proc. of Third Int. Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 202–223. SIAM.

[LWMP96] Lichtenberg B., Webb K. J., Meade D. B., and Peterson A. F. (1996) Comparison of two-dimensional conformal local radiation boundary conditions. *Electromagnetics* 16: 359–384.

52

On the Use of Multigrid as a Preconditioner

C.W. Oosterlee and T. Washio

1 Introduction

In the search for robust and efficient Krylov subspace methods, multigrid is being considered as a preconditioner. With preconditioners based on multigrid it is expected that robust convergence can be achieved for a large class of problems. GMRES [SS86] is used as the Krylov subspace solver. Singularly perturbed 2D problems of both convection-diffusion and of jumping coefficients type are considered, for which the design of optimal standard multigrid is not easy. For these problems the multigrid method is being compared as a solver and as a preconditioner. Eigenvalue spectra of the multigrid iteration matrix are analyzed to understand the convergence of the algorithms.

In the domain decomposition context we can think of the method as a robust subdomain solver. Also, the parallel multiblock method can be seen as an alternative for domain decomposition techniques on regular grids: The method is parallelized with *grid partitioning* [MFL$^+$91].

The purpose of this work is not to derive optimal multigrid methods for specific problems, but to construct a robust well-parallelizable solver, in which the smoother as well as the coarse grid correction is fixed. Another robust multigrid variant for solving scalar partial differential equations is Algebraic Multigrid (AMG) by Ruge and Stüben [RS87], in which the smoother is fixed but the transfer operators depend on the connections in a matrix. Efficient parallelization of AMG is, however, not trivial. Matrix-dependent transfer operators are employed so that problems with convection dominance, as well as problems with jumping coefficients, can be solved efficiently. The operators have been designed so that problems on grids with arbitrary mesh sizes, not just powers of two (+1), can be solved with similar efficiency. The algorithm uses the prolongation operators introduced by de Zeeuw [Zee90]. The multigrid algorithm employs Galerkin coarsening [Hac85], [Wes92] for building the matrices on coarser grids. The alternating zebra line Gauss-Seidel relaxation method is used as the smoother, since it is a robust smoother for anisotropic problems and it is efficiently parallelizable. In [Ket82] an early comparison of multigrid and multigrid

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

preconditioned CG for symmetric model equations showed the promising robustness of the latter method.
The solution method is analyzed in order to understand the convergence behavior of multigrid used as a solver and as a preconditioner. The eigenvalue spectrum of the multigrid iteration matrix for the singularly perturbed problems is calculated in Section 3. Interesting subjects for the convergence behavior are the spectral radius and the eigenvalue distribution. Numerical results are also presented in Section 3. The benefits of the constructed multigrid preconditioned Krylov methods are shown for a convection-diffusion problem and a problem with jumping coefficients on fine grids solved on the NEC Cenju–3 MIMD machine [HCH$^+$96]. The message-passing is done with MPI.

2 The Multigrid Preconditioned Krylov Methods

We concentrate on linear matrix systems with nine diagonals,

$$A\phi \quad = \quad b \ . \tag{2.1}$$

Matrix A has right preconditioning as follows:

$$AK^{-1}(K\phi) = b \ . \tag{2.2}$$

The Krylov subspace method that is used for solving (2.2) is GMRES(m) [SS86]. Matrix K^{-1} in (2.2) is approximated by one iteration of the multigrid method.
Using a preconditioner as solver. A preconditioner, like the multigrid preconditioner, is a candidate for use as a solver. An iteration of a multigrid solver is equivalent to a Richardson iteration on the preconditioned matrix. With K being the iteration matrix, multigrid can be written as follows:

$$K\phi^{(k+1)} + (A - K)\phi^{(k)} = b \ . \tag{2.3}$$

This formulation is equivalent to,

$$\phi^{(k+1)} = \phi^{(k)} + K^{-1}(b - A\phi^{(k)}) = \phi^{(k)} + K^{-1}r^{(k)}; \ \ r^{(k+1)} = (I - AK^{-1})r^{(k)} \ . \tag{2.4}$$

The multigrid solver is implemented as a Richardson iteration with a left multigrid preconditioner for A. The convergence of (2.4) can be investigated by analyzing the spectrum of the iteration matrix. The spectral radius of $I - AK^{-1}$ determines the convergence. This spectrum is analyzed in Section 3 for the multigrid method for the problems tested on 33^2 and 65^2 grids. With this spectrum we can also investigate the convergence of GMRES, since the spectra of left and right preconditioned matrices are the same.
The multigrid preconditioner. The multigrid preconditioner implemented is now discussed in some more detail. The multigrid correction scheme [Hac85, Wes92] is used for solving the linear equations. Here, the robustness and efficiency of the F-cycle is evaluated. The multigrid F-cycle is a hybrid between the cheap V-cycle and the expensive W-cycle. The smoother is the alternating zebra line Gauss-Seidel smoother.

First, all odd (white) lines are processed in one direction, after which all even (black) lines are processed. This procedure takes place in the x- and y-directions. Fourier smoothing analysis for model equations, presented in [Wes92], indicates the robustness of this smoother. The algorithm evaluated adopts the "upwind" prolongation operator by de Zeeuw [Zee90]. This operator is specially designed for problems with jumping coefficients and for second-order differential equations with a dominating first-order derivative. As already indicated in [Den83] it is appropriate for the construction of transfer operators for unsymmetric matrices to split a matrix A into a symmetric and an antisymmetric part:

$$S = \frac{1}{2}(A + A^T), \;\; T = A - S = \frac{1}{2}(A - A^T) \;\; . \tag{2.5}$$

The investigated transfer operators are also based on this splitting. Analysis of this operator and the numerical experiments in [Zee90] shows the very interesting behavior of these operators. Restriction R^L is defined as the transpose of the prolongation operator. The coarse grid matrices A^L are defined with Galerkin coarsening [Hac85], [Wes92],

$$\begin{aligned} A^M &= A \; , & (2.6)\\ A^L &= R^L A^{L+1} P^{L+1}, \; 1 \leq L \leq M-1 \; . & (2.7) \end{aligned}$$

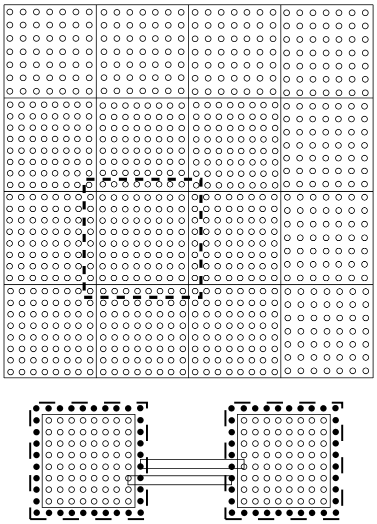
M represents the finest grid level.
Grid partitioning. If grid applications are to be implemented on parallel computers, a straightforward approach is to use ***grid partitioning.*** This means that the original domain Ω is split into P parts (subdomains) in such a way that, with respect to the finest grid, each subdomain consists of (roughly) the same number of grid points. Because of the only local dependencies of grid points, each process needs foreign data only from boundary areas of neighbor subdomains. After a smoothing step is performed, data have to be communicated along the artificial boundaries (see Figure 1). The extension of the single grid case to parallel multigrid is obvious: On the finest grid level, all communication is a strictly local one. Similarly, also on all coarser grids necessary communication is "local" relatively to the corresponding grid level.

Parallelism is straightforward in Krylov methods, except for the multigrid preconditioner, the matrix-vector and inner products, which need communication among neighboring processors for the problems under consideration. We would like to point out that in our approach all parallel algorithms are algorithmically equivalent to their non–partitioned versions: the results of the partitioned and the non–partitioned versions are identical.

3 Numerical Results

The equations investigated are two 2D singularly perturbed problems. A convection-diffusion equation with a dominating convection term and an interface problem are solved. We concentrate on "difficult" problems for multigrid. As the initial guess $\phi^{(0)} = 0$ is used for all problems. Restart parameter m is set to 20 here. After some efficiency tests, we choose no pre-smoothing and 2 post-smoothing iterations;

Figure 1 A regular grid partitioned into 16 subgrids. To each subgrid an overlap area is assigned needed for data exchange in the exchange phase.



on the coarsest grid 2 smoothing iterations are performed ($\nu_3 = 2$) in order to keep the parallel method as cheap as possible. For the problems investigated this did not influence convergence negatively, since the coarsest grid is always a 3^2 grid. The results presented are the number of iterations (n) to reduce the L_2-norm of the initial residual with 8 orders of magnitude ($||r^{(n)}||_2/||r^{(0)}||_2 \leq 10^{-8}$). Furthermore, the elapsed time for this number of iterations obtained on the NEC Cenju–3 MIMD machine [HCH$^+$96] is presented. For all problems 32 processors are used in a 4 × 8 configuration.

Rotating convection-diffusion equation. The first problem is a rotating convection-diffusion problem,

$$-\epsilon\Delta\phi + a(x,y)\frac{\partial\phi}{\partial x} + b(x,y)\frac{\partial\phi}{\partial y} = 1 \ \text{ on } \Omega = (0,1)\times(0,1) \ . \tag{3.8}$$

Here, $a(x,y) = -\sin(\pi x).\cos(\pi y)$, $b(x,y) = \sin(\pi y).\cos(\pi x)$

Dirichlet boundary conditions are prescribed: $\phi|_\Gamma = \sin(\pi x) + \sin(13\pi x) + \sin(\pi y) + \sin(13\pi y)$.

A convection dominated test case is chosen: $\epsilon = 10^{-5}$. The convection terms are discretized with a standard upwind discretization. A first order upwind discretization is chosen, since this is still a linear discretization, which can be tested and evaluated. The final target is a second order (nonlinear) discretization with a limiter, for which the components chosen here in multigrid (and the linear GMRES solver) are not appropriate. However, it is a useful discretization for understanding the behavior of our multigrid as a preconditioner and as a solver. We investigate the eigenvalue spectrum of the Richardson iteration matrix (2.4) on a 33^2 and a 65^2 grid. The spectra are presented in Figure 2. As can be seen, most eigenvalues are clustered around 0, only the largest eigenvalues are outside the clustering. The spectral radius determines the convergence of multigrid as a solver, as is well-known. This spectral radius increases on finer grids as can be seen in Figure 2. However, it is found that the eigenvectors belonging to the larger eigenvalues are very soon captured into the Krylov subspace when multigrid is used as a preconditioner, and that therefore the convergence is accelerated considerably.

Figure 2 *The eigenvalue spectra for the rotating convection-diffusion problem on two consecutive grid sizes.*

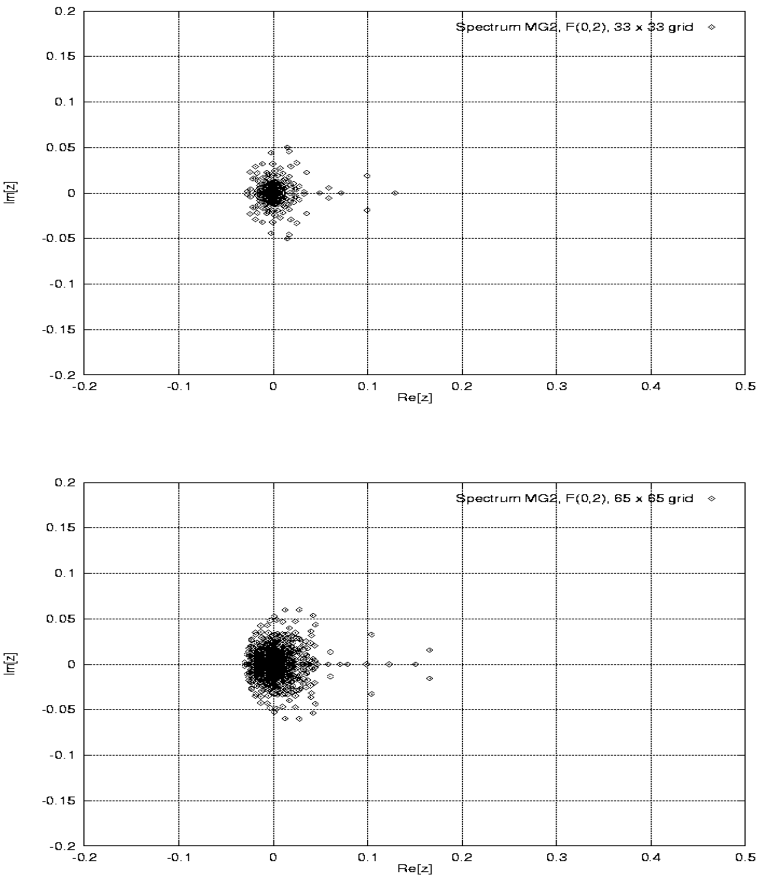


Table 1 presents the convergence results of multigrid as a solver and as a preconditioner on three different grid sizes, 129^2, 257^2 and 513^2. It can be seen that multigrid used as preconditioner handles this test case with dominating convection very well. Very satisfactory convergence associated with small elapsed times is found in most cases with the F-cycle. (With the V-cycle used as a preconditioner the best elapsed times are found, but the number of iterations is increasing with a higher degree than with the F-cycle for large grid sizes.)

An interface problem. Next, an interface problem is considered. This type of problems has been investigated with multigrid, for example in [Zee90]. The problem to be solved

Table 1 *Iterations (n) and elapsed time in seconds for the rotating convection-diffusion equation.*

grid: method:	129^2	257^2	513^2
multigrid as solver	(15) 5.1	(20) 10.1	(29) 24.8
multigrid as preconditioner	(10) 3.5	(12) 6.1	(16) 14.3

looks as follows:

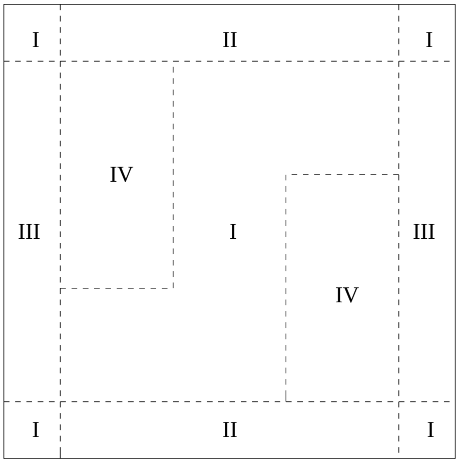
$$\frac{\partial}{\partial x} D_1 \frac{\partial \phi}{\partial x} + \frac{\partial}{\partial y} D_2 \frac{\partial \phi}{\partial y} = 1 \text{ on } \Omega = (0,1) \times (0,1) \ . \tag{3.9}$$

Dirichlet conditions are used:

$$\phi = 1 \quad \text{on } \{x \leq \frac{1}{2} \wedge y = 0\} \text{ and on } \{x = 0 \wedge y \leq \frac{1}{2}\}; \text{ elsewhere } \phi = 0. \tag{3.10}$$

The computational domain with the values of the jumping diffusion coefficients D_1 and D_2 is presented in Figure 3. The discretization is vertex-centered and all diffusion coefficients are assumed in the vertices. For the discretized diffusion coefficient between two vertices the harmonic average of the neighboring coefficients is taken.

Figure 3 The domain for the interface problem



I: $D_1 = 10^{-3}$, $D_2 = 10^{-3}$

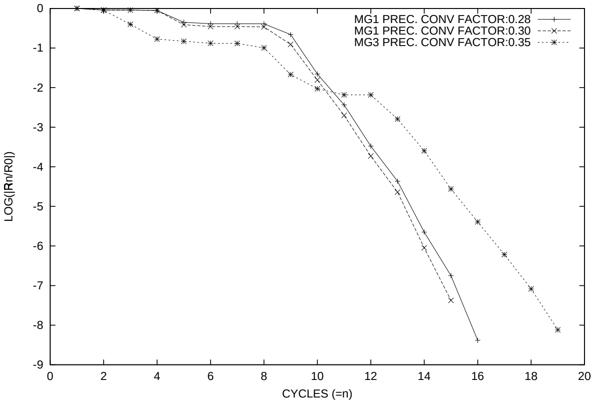
II: $D_1 = 10^{3}$, $D_2 = 10^{-3}$

III: $D_1 = 10^{-3}$, $D_2 = 10^{3}$

IV: $D_1 = 1$, $D_2 = 1$.

Clearly multigrid can solve many interface problems, presented for example in [Zee90]. Here we constructed a difficult problem, where the robust components of our multigrid method are not satisfactory. The Krylov acceleration is really needed for convergence. The eigenvalue spectrum obtained with multigrid is presented in Figure 5. In Figure 5 we see two eigenvalues close to 1, so multigrid convergence is already very slow on this coarse grid. The convergence of GMRES(20) with multigrid as a preconditioner on the 33^2 grid is shown in Figure 4. Multigrid as a preconditioner is

Figure 4 The convergence of GMRES(30) with multigrid preconditioner, 33^2 grid.



converging well. In Figure 4 it can be seen that for the problem on a 33^2 grid the first 9 GMRES iterations do not reduce the residual very much, but after iteration 9 fast convergence is obtained. In our analysis of the evolution of the Krylov subspace it is found that the vector belonging to a second eigenvalue of $I - AK^{-1}$ around 1 is obtained in the Krylov space in the 9th iteration, and then GMRES starts converging very rapidly. For this test problem the convergence of the preconditioned Krylov methods with the multigrid preconditioner on three very fine grids is presented. The number of GMRES(20) iterations (n) and the elapsed time are presented in Table 2. The GMRES convergence is influenced by the fact that the restart parameter is 20; a larger parameter results in faster convergence. Again the F-cycle is preferred for its robustness and efficiency.

Figure 5 The eigenvalue spectrum for the interface problem on a 33^2 grid, F(0,2) cycle.

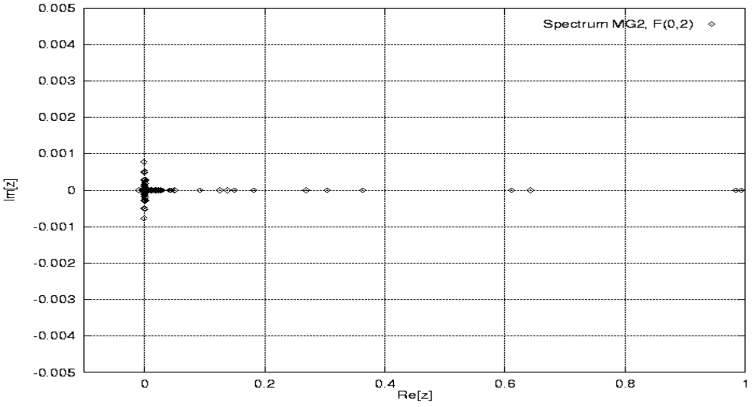


Table 2 *GMRES(20) iterations (n) and elapsed time in seconds for the interface problem.*

grid:	257^2	513^2	769^2
cycle:	GMRES	GMRES	GMRES
F	(34) 19.0	(33) 30.6	(36) 52.9

4 Conclusion

In the present work a multigrid method has been evaluated as a solver and as a preconditioner for GMRES. The problems investigated were singularly perturbed. The behavior of the multigrid method is much more robust when it is used as a preconditioner, since then the convergence is not sensitive to parameter changes. For the test problems many of the eigenvalues of a multigrid iteration matrix are clustered around the origin. In some cases there are some isolated large eigenvalues which limit the multigrid convergence, but are captured nicely by a Krylov acceleration technique. The most efficient results are obtained when the method is used as a preconditioner. The multigrid F-cycle is used, since it is robust and efficient. The convergence behavior can be well understood by investigating the eigenvalue spectrum of the iteration matrix.

REFERENCES

[Den83] Dendy Jr. J. (1983) Black box multigrid for nonsymmetric problems. *Appl. Math. Comp.* 13: 261–283.

[Hac85] Hackbusch W. (1985) *Multi-grid methods and applications.* Springer, Berlin.

[HCH$^+$96] Hempel R., Calkin R., Hess R., Joppich W., Keller U., Koike N., Oosterlee C., Ritzdorf H., Washio T., Wypior P., and Ziegler W. (1996) Real applications on the new parallel system NEC Cenju–3. *Par. Comput.* 22: 131–148.

[Ket82] Kettler R. (1982) Analysis and comparison of relaxation schemes in robust multigrid and preconditioned conjugate gradient methods. In Hackbusch W. and Trottenberg U. (eds) *Multigrid Methods*, volume 960 of *Lect. Notes in Math.*, pages 502–534. Springer, Berlin.

[MFL$^+$91] McBryan O., Frederickson P., Linden J., Schüller A., Solchenbach K., Stüben K., Thole C., and Trottenberg U. (1991) Multigrid methods on parallel computers - a survey of recent developments. *IMPACT Comp. Science Eng.* 3: 1–75.

[RS87] Ruge J. and Stüben K. (1987) Algebraic multigrid (AMG). In McCormick S. (ed) *Multigrid Methods*, volume 5 of *Frontiers in Appl. Math.*, pages 73–130. SIAM Press, Philadelphia.

[SS86] Saad Y. and Schultz M. (1986) GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Comput.* 7: 856–869.

[Wes92] Wesseling P. (1992) *An introduction to multigrid methods.* John Wiley, Chichester.

[Zee90] Zeeuw de P. (1990) Matrix-dependent prolongations and restrictions in a blackbox multigrid solver. *J. Comp. Appl. Math.* 33: 1–27.

53

Newton-Krylov Domain Decomposition Solvers for Adaptive *hp* Approximations of the Steady Incompressible Navier-Stokes Equations with Discontinuous Pressure Fields

Abani Patra

1 Introduction

In this paper, we extend non-overlapping domain decomposition techniques, previously developed for second order elliptic problems and the Stokes operator [LP96], to the solution of incompressible flow problems governed by the Navier-Stokes equations. In this approach, we will reduce the original problem to a problem set on the subspace of divergence free functions, and apply existing domain decomposition techniques to the resulting sub-problem. The advantage of this approach is to greatly reduce the size of the algebraic systems that have to be solved.

Adaptive *hp* finite elements, in which the spectral order and element size are independently varied over the whole domain, are capable of delivering solution accuracies far superior to classical $h-$ or $p-$version finite element methods, for a given discretization size. Several researchers [BS87, DORH89, ROD89] have, in fact, shown that the reduction in discretization error with respect to number of unknowns can be exponential for general classes of elliptic boundary value problems, as opposed to the asymptotic algebraic rates observed for h or p-version finite element methods. Together with multiprocessor computing, these methods thus offer the possibility of orders-of-magnitude improvement in computing efficiency over existing finite element models.

The principal computational cost in any finite element solution is encountered in the solver. For the nonlinear Navier-Stokes equations, solved using a Newton iteration scheme, the major computational cost is in the linear solve in each iterate. If time

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

stepping is used to linearize the problem, the use of implicit methods, leads to a similar situation. In a parallel computing environment, conventional direct solvers based on some variant of Gauss elimination are usually inefficient for the irregular sparse linear systems generated by adaptive *hp* discretizations. Further, these linear systems are often very poorly conditioned, ruling out most standard iterative solvers. Thus, efficient solvers, meeting the twin criteria of being parallelizable and controlling the conditioning of the system, need to be used.

In this paper, we discuss a practical and efficient parallel iterative solver that meets the above criteria. The solver combines nonlinear Newton iterations with iterative substructuring and coarse grid preconditioning. The inner solver for the linearized problem at each Newton iterate can be thought of as a combination of multiple direct solvers at the subdomain level together with a preconditioned iterative solver, to handle the interface problem efficiently in parallel. The iterative solver of choice is the GMRES algorithm.

In the remaining sections we introduce the steady state incompressible Navier-Stokes equations and its weak formulation, its finite element discretization and describe a domain decomposition iterative solver and present some numerical results.

2 The Steady Navier-Stokes Equations

We define the spaces $V = (H_0^1(\Omega))^2$ and $Q = L_0^2(\Omega)$ and a domain $\Omega \subset I\!\!R^2$ with boundaries $\partial\Omega$ that are assumed to be locally Lipschitz . The Navier-Stokes problem on $\Omega \subset I\!\!R^2$ consists of finding a velocity, pressure pair $(u, P) \in V \times Q$ satisfying

$$\begin{aligned} (u \cdot \nabla)u - \nu\Delta \cdot u + \nabla P &= f \quad in\ \Omega \\ \nabla \cdot u &= 0 \quad in\ \Omega \\ u &= g \quad on\ \partial\Omega \end{aligned}$$

where f is a body force, ν is the kinematic viscosity.

Basic Formulation and LBB condition

Let us consider the following mixed finite element approximation of the Navier-Stokes problem:

Find $u_h \in V_h$, $p_h \in W_h$ such that

$$c(u_h, u_h, v_h) + a(u_h, v_h) + \quad b(v_h, p_h) = L(v_h) \qquad \forall v_h \in V_h \tag{2.1}$$

$$b(u_h, q_h) = -b(\bar{u}, q_h) \qquad \forall q_h \in W_{h0} \tag{2.2}$$

where $u_h + \bar{u}$ is the approximate velocity field and p_h is the approximate pressure field inside an incompressible viscous fluid flowing through a given domain $\Omega \subset I\!\!R^2$ with imposed velocity $\bar{u}$ at the boundary $\partial\Omega$ of Ω. The domain Ω is partitioned into finite elements such that $\bar{\Omega} = \cup_e K_e$, and the finite element spaces V_h and W_{h0} are conforming finite dimensional approximations of $H_0^1(\Omega, I\!\!R^2)$ and $L_0^2(\Omega)$ given by

$$\begin{array}{ll} V_h & = \{v_h \in C(\bar{\Omega}, I\!\!R^2), v_{h|K_e} \in Q_k(K_e), \forall e, v_h = 0 \text{ on } \partial\Omega\}, \\ W_h & = \{q_h \in L^2(\Omega), q_{h|K_e} \in Q_l(K_e), \forall e\}, \text{ and } W_{h0} = \{q_h \in W_h, \int_\Omega q_h d\Omega = 0\}. \end{array}$$

In this framework the trilinear form c, the bilinear forms a and b and the linear form L are defined by

$$\begin{array}{lll} c(u,u,v) & = \int_\Omega (u \cdot \nabla)u.v d\Omega, & \forall u, v \in H^1(\Omega, I\!\!R^2) \\ a(u,v) & = \int_\Omega \nu \nabla u.\nabla v d\Omega, & \forall u, v \in H^1(\Omega, I\!\!R^2) \\ b(v,q) & = \int_\Omega q \text{div} v d\Omega, & \forall v \in H^1(\Omega, I\!\!R^2) \text{ and } \forall q \in L^2(\Omega), \\ L(v) & = \int_\Omega f.v d\Omega - a(\bar{u}, \bar{v}) - c(\bar{u}, \bar{u}, v) & \end{array}$$

We now introduce a non-overlapping partition of Ω into a finite number of subdomains such that $\bar{\Omega} = \cup_{i=1}^N \bar{\Omega}_i$ The interface $\Gamma_i = \partial\Omega_i - \partial\Omega$ is supposed to coincide with interelement boundaries, and we suppose that the finite element spaces V_h and W_h satisfy the following LBB compatibility condition on each subdomain(including $\Omega_0 = \Omega$).
Assumption 1 There exists a constant $\beta(k)$ independent of the mesh size h, but possibly dependent on the local degree k of the finite elements, such that

$$\inf_{q \in L_0^2(\Omega_i) \cap W_h} \sup_{v \in H_0^1(\Omega_i) \cap V_h} \left[\frac{\int_{\Omega_i} q \text{ div} v \, d\Omega}{||q||_{0,\Omega_i} ||v||_{1,\Omega_i}} \right] \geq \beta(k) \; \forall i = 0, ..., N \tag{2.3}$$

3 Choice of Compatible Spaces for Adaptive *hp* FEM

We begin the discussion by defining appropriate polynomial spaces. Let $L_i(t)$ denote the Legendre polynomial of degree i. Now define $U_i(x) = \int_{-1}^x L_i(t)dt$ the integrated Legendre polynomial. Note that $U(\pm 1) = 0$. Further define the "volumetric" and "lateral" polynomial combinations over the master element $\hat{K} = \hat{I}_x \times \hat{I}_y = [-1,1] \times [-1,1]$ as

$$J_k(\hat{K}) = \{\sum_{i,j=1}^{k-1} a_{ij} U_i(x) U_j(y), a_{ij} \in I\!\!R\} \text{ and } E_k(\hat{K}) = P_1(\hat{I}_x)P_k(\hat{I}_y) \cup P_k(\hat{I}_x)P_1(\hat{I}_x)$$

where $P_k(\hat{I}_x)$ (resp. $P_k(\hat{I}_y)$) denotes the space of polynomials of maximum degree k in x (resp. y). Different choices of finite elements for an hp approximation may now be defined by appropriate combinations of E_k and J_k for each element. For example, one may choose $Q_k(\hat{K}) = E_k(\hat{K}) \oplus J_k(\hat{K})$.

In the context of p version finite element methods, Stenberg and Suri [SS95] have recently proposed systematic ways of constructing compatible higher order approximations for the velocity and pressure spaces such that Assumption 1 is satisfied automatically. One such construct is $W_k(\hat{K}) = Q_{k-1}(\hat{K})$, with corresponding velocity space $V_k(\hat{K}) = Q_{k+1}(\hat{K}) \; \forall k \geq 2$.

In adaptive p and hp methods, the polynomial order may change from element to element with C^0 continuity of the velocity approximation being maintained by extending the higher order function on the shared edge into the lower order element. The space $E_k(\hat{K})$ must be redefined to reflect this.

For $k1, k2, k3, k4 \geq 2$

$$\begin{aligned} E_{\mathbf{k}}(\hat{K}) &= E_{k1}(\gamma_1) \oplus E_{k2}(\gamma_2) \oplus E_{k3}(\gamma_3) \oplus E_{k4}(\gamma_4) \\ &= \mathcal{P}_{k1}(\gamma_1)\mathcal{P}_1(y) \oplus \mathcal{P}_{k2}(\gamma_2)\mathcal{P}_1(x)\mathcal{P}_{k3}(\gamma_3)\mathcal{P}_1(y) \oplus \mathcal{P}_{k4}(\gamma_4)\mathcal{P}_1(x) \end{aligned}$$

where $\mathbf{k} = \{k_1, k_2, k_3, k_4\}$ and E_{ki} are the edge spaces of polynomial order ki defined on the edges γ_i. Let $k_m = max\{k_1, k_2, k_3, k_4\}$. Now the spaces for the adaptive *hp*-version can be redefined for this situation as

$$\begin{aligned} W_k(\hat{K}) &= E_{k1-1}(\gamma_1) \oplus E_{k2-1}(\gamma_2) \oplus E_{k3-1}(\gamma_3) \oplus E_{k4-1}(\gamma_4) \oplus J_{k_m-1}(\hat{\omega}) \\ V_k(\hat{K}) &= E_{\mathbf{k}} \oplus J_{k_m+1}(\hat{\omega}). \end{aligned}$$

This is easily implemented in the code by augmenting the order of the bubble functions for velocity to one more than the maximum of all the edge polynomial orders. The pressure shape functions are then constructed as one order less than the velocity functions on the edges and two orders less in the bubble function. As show in [LP96] Assumption 1 can be established for this construction of the spaces.

4 Solution Algorithm

The Nonlinear Iteration

For flows characterized by low Reynolds numbers ($Re = 1/\nu \approx 100$) it is often easy and possible to solve the nonlinear system of equations arising from an adaptive *hp* discretization of (2.1,2.2) using Newton's method. However, for higher Reynolds numbers this technique does not appear to perform well. Newton's method generates a sequence of iterates of the form $u_h^k, k = 0, 1, 2, ...$ Given u_h^{k-1} we find u_h^k by solving

$$\left.\begin{array}{lll} c(u_h^k, u_h^{k-1}, v_h) + & c(u_h^{k-1}, u_h^k, v_h) & + a(u_h^k, v_h) \ + \ b(v_h, p_h^k) = (f, v_h) + \\ & & c(u_h^{k-1}, u_h^{k-1}, v_h) \ \forall v_h \in V_h \\ b(u_h^k, q_h) & & = 0 \qquad\qquad \forall q_h \in W_{h0} \end{array}\right\} \qquad (4.4)$$

Thus at each stage we need to solve a large irregularly sparse linear system. We will now discuss an efficient parallel solver for this system. The solver uses iterative substructuring with coarse grid preconditioning of the type discussed in [OPF94, LP96].

Reduction to Interface Problem

We use fast local subdomain based sparse solvers and the discontinuity of the pressure field to reduce the global problem to one posed purely in terms of the interface velocities. We start by using the structure in the p version to decompose the velocity space at the element level into the three subspaces of vertex functions (V), edge functions (E) and bubble functions (B), and augment it with a pressure space(P). The element stiffness can then be written as:

$$\mathbf{K_{elt}} = \begin{bmatrix} VV_{elt} & VE_{elt} & VP_{elt} & VB_{elt} \\ EV_{elt} & EE_{elt} & EP_{elt} & EB_{elt} \\ PV_{elt} & PE_{elt} & 0 & PB_{elt} \\ BV_{elt} & BE_{elt} & BP_{elt} & BB_{elt} \end{bmatrix} \begin{Bmatrix} u_V \\ u_E \\ P \\ u_B \end{Bmatrix}$$

We also compute a vector corresponding to $b(u_h, 1) = \bar{B}_{el}$ for subsequent use in the preconditioning form.

The bubble functions have support only inside an element, so they may be immediately eliminated using a static condensation procedure. This modifies the rest of $\mathbf{K_{elt}}$ and $\bar{B}_{el}$. The zero on the diagonal corresponding to the pressure degrees of freedom is now replaced by $\widehat{PP} = -(PB)(BB)^{-1}(BP)$. If the pressure is assumed to be continuous in each subdomain and discontinuous across interfaces (Γ_i) we obtain a subdomain stiffness matrix of the form:

$$\mathbf{K_I} = \begin{bmatrix} WW_I & WF_I & WI_I \\ FW_I & FF_I & FI_I \\ IW_I & IF_I & II_I \end{bmatrix} \qquad \mathbf{II_I} = \begin{bmatrix} \widetilde{VV_I} & \widetilde{VE_I} & \widetilde{VP_I} \\ \widetilde{EV_I} & \widetilde{EE_I} & \widetilde{EP_I} \\ \widetilde{PV_I} & \widetilde{PE_I} & \widetilde{PP_I} \end{bmatrix}$$

where WW_I and FF_I denote the vertex and edge degrees of freedom associated with subdomain interfaces on the I^{th} subdomain and II_I denotes those on the interior.

If the pressure field is assumed to be discontinuous across elements then, the pressure degrees of freedom may be eliminated at the element level with respect to an average element pressure. This elimination is carried out using a procedure identical to the static condensation used on the bubble nodes after setting one of the pressure nodes to a value of zero. This is necessary to make $\widehat{PP}$ invertible. The pressures can be computed consistently by requiring $p_{ih} \in L^2_0(\Omega_i)$. If the actual pressure on this node is denoted p, then the actual pressure is $p = \bar{P}_{el} + P_{rel}$ where, $P_{rel} \in L^2_0(\Omega_i)$ is the relative pressure computed. The value $\bar{P}_{el}$ of the pressure on the remaining node can be computed from the velocities at the subdomain level and corresponds therefore to a subdomain internal degree of freedom. It will be associated to the the local constraint of volume conservation $\int_{elt}$ div $u_{ih} = 0$. These values can be eliminated by treating this local constraint by a penalty approach on all subdomain elements except one.

The static condensation process can now be used (irrespective of the pressure approximation) at the subdomain level to obtain

$$\widetilde{\mathbf{K_I}} = \begin{bmatrix} \widetilde{WW}_I & \widetilde{WF}_I & 0 \\ \widetilde{FW}_I & \widetilde{FF}_I & 0 \\ 0 & 0 & II_I \end{bmatrix} \qquad \mathbf{S_I} = \begin{bmatrix} \widetilde{WW}_I & \widetilde{WF}_I \\ \widetilde{FW}_I & \widetilde{FF}_I \end{bmatrix}$$

Note that the same modifications are also carried out on $\bar{B}_I$.

The matrix S_I is the contribution of each subdomain to the interface operator S. The vectors B_I assemble into $\bar{B}$ described in conjunction with the solution of the interface operator. Procedures for parallel iterative solution for S are discussed next.

Interface Solver using GMRES and Divergence-Free Search Vectors

The interface operator S is non-symmetric and the iterative solver of choice is usually GMRES, a method that minimizes the residual over a Krylov space. The basic GMRES algorithm however suffers from the the drawback that the work grows quadratically and storage grows linearly with the number of iterations. The restarted version of the algorithm alleviates this difficulty to some extent, and is the one used in this study. The preconditioned version of the GMRES includes in each iteration, a solve $\mathbf{C}$ $G^n = R^n$ where C is a preconditioning operator, R^n is the residual in the n^{th} iterate and G^n is the computed search vector.

Further, to satisfy the incompressibility condition on the interface velocities, we restrict our choice of search directions to divergence free vectors. This is accomplished by modifying the preconditioning step $\mathbf{C}$ $G^n = R^n$ to

$$\begin{aligned} CG^n + \quad \bar{B}^T\bar{p} &= \quad g \\ \bar{B}G^n &= \quad 0 \end{aligned}$$

where

$$\bar{B} = \int_{\Omega_i} \nabla v_h.1 \; d\Omega_i = \int_{\Gamma_i} v_h.n \; d\Gamma$$

and $\bar{p}$ is a vector of average pressure per subdomain. This computation reduces to one coarse solve of a problem of dimension equal to the number of subdomains per application of the preconditioner, and the initial cost of setting up and factoring $\bar{B}C^{-1}\bar{B}^T$.

Choice of Preconditioner

As described in [OPF94, LP96] matrix S is naturally blocked into a small portion ($\widetilde{WW}$) corresponding to the nodes on the interface and the larger portion corresponding to the unknowns associated with the edges($\widetilde{FF}$) and their interactions $\widetilde{WF}$ and $\widetilde{FW}$. As a preconditioner C, we explore two choices, denoted C_1 and C_2, analogous to the choices in [LP96]. These are a) the $\widetilde{WW}$ block and the diagonals of $\widetilde{FF}$ and b) the $\widetilde{WW}$ block and the block diagonals of $\widetilde{FF}$. Block diagonals correspond to the degrees of freedom associated with a particular edge. In matrix notation these are:

$$\mathbf{C_1} = \begin{bmatrix} \widetilde{WW}_I & 0 \\ 0 & diag(\widetilde{FF}_I) \end{bmatrix} \qquad \mathbf{C_2} = \begin{bmatrix} \widetilde{WW}_I & 0 \\ 0 & diagB(\widetilde{FF}_I) \end{bmatrix}$$

where $diag(\widetilde{FF}_I)$ and $diagB(\widetilde{FF}_I)$ denote the diagonal and block diagonal respectively.

Figure 1 Driven Cavity Flow for low Reynolds numbers. Solution obtained using two level Newton-Krylov domain decomposition solver.

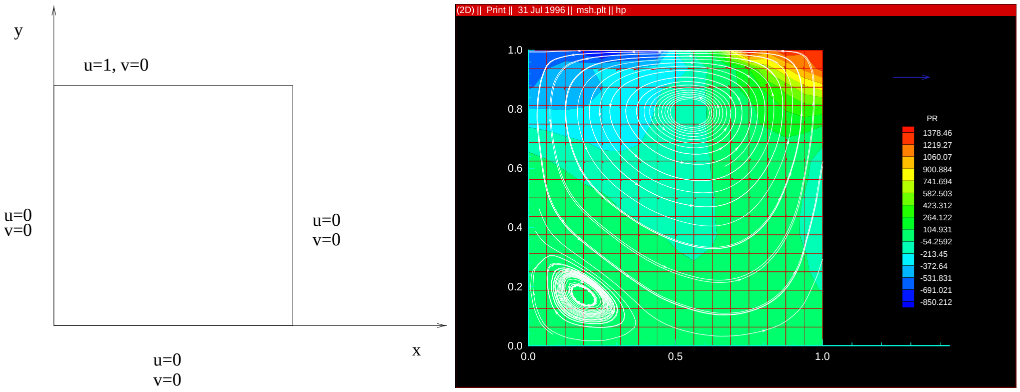
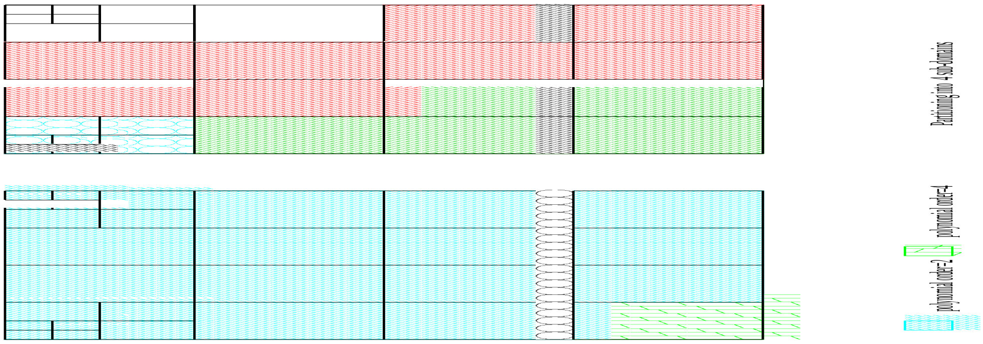


Figure 2 Sample *hp* adapted mesh for the problem and partitioning into 4 subdomains. Solver converged in 9 preconditioned GMRES and 6 Newton iterations.

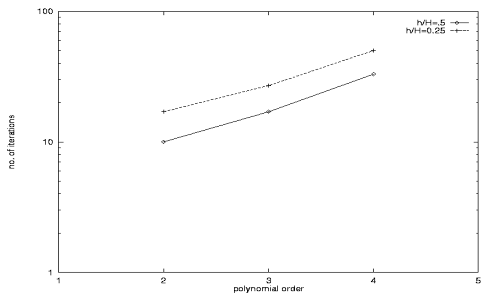


5 Numerical Results

The algorithm described in this paper is new and only validation studies for low Reynolds number ($\frac{1}{\nu} \approx 100$) are currently available. Numerical experience indicates a strong dependence on the highest polynomial order used and a weaker dependence on the smallest element size used. Fig. 1 shows the test problem and results obtained on a uniform discretization of 64 quadratic elements using this solver. Fig. 2 shows a sample *hp* adaptive mesh and its partitioning into 4 subdomains, also used on the same problem. The GMRES algorithm used in the inner loop for the linearized problem shows good convergence (see Fig. 3) for polynomial orders $p \leq 4$ and values of $h/H \geq .125$, where h/H is the ratio of mesh size to subdomain size.

The use of the two level iterative scheme permits us to use inexact solves in the linear solver and still obtain fairly rapid convergence of the overall solution algorithm. This option needs further testing to establish minimum levels of accuracy in the inner loop to maintain convergence rates in the outer loop. Experience seems to indicate that between 10 and 15 GMRES iterations are adequate.

Figure 3 Convergence rates obtained for inner loop GMRES solver using C_1 preconditioner, for different choices of h/H (ratio of minimum element size to subdomain size) and polynomial order p.



Acknowledgement

The support of this work by NSF under grant no. ASC-9504075 is gratefully acknowledged. The main part of this paper was drafted while the author was a visiting scholar at TICAM.

REFERENCES

[BS87] Babuska I. and Suri M. (1987) The *h-p* version of the finite element method with quasiuniform meshes. *Mathematical Modelling and Numerical Analysis* 21(2): 199–238.

[DORH89] Demkowicz L., Oden J. T., Rachowicz W., and Hardy O. (1989) Toward a universal *hp* adaptive finite element strategy, part 1. constrained approximation and data structure. *Comput. Methods. Appl. Mech. and Engg.* 77: 79–112.

[LP96] LeTallec P. and Patra A. (1996) Non-overlapping domain decomposition methods for adaptive *hp* approximations of the stokes problem with discontinuous pressure fields. TICAM Report 33, University of Texas-Austin, Austin, TX-78712. to appear in *Comput. Methods. Appl. Mech. and Engg.*

[OPF94] Oden J. T., Patra A., and Feng Y. S. (1994) Parallel domain decomposition solver for adaptive *hp* finite element methods. TICAM Report 11, University of Texas-Austin, Austin, TX-78712. to appear in *SIAM Journal for Numerical Analysis.*

[ROD89] Rachowicz W., Oden J., and Demkowicz L. (1989) Toward a universal *hp* adaptive finite element strategy; part 3: Design of *hp* meshes. *Comput. Meth. Appl. Mech. and Engg.* 77: 181–212.

[SS95] Stenberg R. and Suri M. (1995) Mixed *hp* finite elements for problems in elasticity and stokes flow. Technical Report 35, Dept. of Math. and Stat, University of Maryland Baltimore County, Baltimore, Maryland.

54

A Parallel Domain Decomposition Method for Spline Approximation

C. K. Pink, I. J. Anderson, and J. C. Mason

1 Introduction

The work described in this paper arises from the desire to approximate in a practical timescale a large set of discrete data with a spline function defined by B-spline basis functions. The least squares approximation of large sets of data is an important but time consuming problem, which has applications in many fields including those of graphics, image processing and computer aided design. We consider this approximation problem for what we term “uniform” sets of data, and we describe a parallel domain decomposition method for its solution. We use *uniform* in its statistical sense, namely we mean that the data is scattered in such a way that the density of points is fairly constant throughout the domain of approximation, as in a uniform distribution.

In section 2 we consider the general problem of B-spline approximation, and in section 3 we describe briefly a method produced by Anderson [And94, And97] for its efficient solution when considering uniform data sets. In section 4 we consider the problems involved with producing a parallel method based on this serial approach and we discuss the solutions of these problems. Results gained from implementing the parallel algorithm on the KSR-1 [Ken93, Pap93] machine are given and analysed in section 5.

2 The Two-dimensional Approximation Problem

Given a set of m data, (x_k, y_k, f_k), we wish to calculate the bivariate spline function that minimises the sum of the squares of the residuals between the data ordinates and the spline function evaluated at the data abscissae [Cox87]. The standard definition of a bivariate tensor-product spline is,

$$s(x,y) = \sum_{i=1}^{q_x} \sum_{j=1}^{q_y} c_{i,j} N_i(x; \boldsymbol{\lambda}) N_j(y; \boldsymbol{\mu}),$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

where $N_i(x; \boldsymbol{\lambda})$ and $N_j(y; \boldsymbol{\mu})$ are the B-spline basis functions in the x and y dimensions with respect to the knot sets $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ respectively. The residuals of this fit are $e_k = f_k - s(x_k, y_k)$ for $k = 1, 2, \ldots, m$. Using vectors we express the function values and residuals as $\mathbf{f}$ and $\mathbf{e}$ respectively. We wish to calculate the $q_x \times q_y$ coefficients, $\{c_{i,j}\}$, which minimise the residual sum of squares, $E = \mathbf{e}^{\mathrm{T}}\mathbf{e}$.

In order to express the B-splines evaluated at the data points in the form of a matrix $\boldsymbol{A}$, we employ a lexicographical ordering of the basis functions. The (k, ℓ)th element of $\boldsymbol{A}$ is given by $a_{k,\ell} = N_i(x_k; \boldsymbol{\lambda}) N_j(y_k; \boldsymbol{\mu})$, where $\ell = (j-1)q_x + i$. Similarly, the coefficients, $\{c_{i,j}\}$, may be expressed in the form of a vector, $\mathbf{c}$, using the same lexicographical ordering. The system of equations resulting from the minimisation problem,

$$\boldsymbol{A}\mathbf{c} = \mathbf{f},$$

is usually overdetermined and thus requires the use of orthogonal transformations, e.g. Givens rotations, Householder transformations or SVD [GV89], for its solution to be determined.

The compact support property of B-spline basis functions produces a banded matrix $\boldsymbol{A}$. This matrix $\boldsymbol{A}$ has at most $n_x \times n_y$ non-zero elements in each row, where n_x and n_y are the order of the approximation in the x and y dimensions, however these elements are not in consecutive columns of the matrix. In fact, the bandwidth of the matrix [Cox82, HH74] is

$$(n_y - 1)q_x + n_x.$$

Consequently the solution of this system can be computed in $\mathcal{O}(mq_x^2 n_y^2)$ [Cox82] floating point operations (flops). Therefore the solution time of the system of equations is dependent on both the order of the spline fit (in the y dimension) and the number of coefficients (in the x dimension).

The dependence of the flop count on the square of the number of coefficients is the important factor with regards to the solution time. To approximate large sets of data well, a comparatively large set of knots is needed, and as the size of the data set increases so we need to increase the size of the knot set. Consequently the linear system that needs to be solved to find the coefficients of the approximation becomes too large to be solved in a practical time.

3 Fast Serial B-spline Approximation

Anderson [And94, And97] has developed a serial method that efficiently approximates large uniform data sets. The basic premise behind the method is to convert an approximation problem on a large domain with a large data set into a set of smaller problems that may be solved more rapidly.

As an example, consider the general least squares approximation problem, described in section 2, on a rectangular domain. The matrix A produced by the system will have m rows, and bandwidth $(n_y - 1)q_x + n_x$, and takes $\mathcal{O}(mq_x^2 n_y^2)$ flops to solve. Now consider this problem as a grid of 10×10 subproblems. Each subproblem produces a matrix A with approximately $m/100$ rows (because of the uniform distribution of the data) and a bandwidth of about $(n_y - 1)q_x/10 + n_x$. This system can be solved in approximately $1/10,000$ of the time taken for the original system, and therefore solving the 100 subproblems will take about 1/100 of the time taken for the original system. Clearly, there are time savings that can be made by exploiting this fact.

Simply subdividing the coefficient array and finding these coefficients from subdomains does not, however, produce the least-squares fit for the overall domain. The supports of adjacent B-spline basis functions of order n overlap by $n-1$ knot intervals. Therefore the regions of support for neighbouring subsets of coefficients also overlap by this amount. This means that many data points will be in more than one region and their values will contribute twice to the overall approximation (a problem that is called 'overfitting' [And94]). To overcome this, after each subdomain approximation has been formed, the coefficients found are added to the overall approximation,

$$c_{i,j}^{(r)} = c_{i,j}^{(r-1)} + \hat{c}_{i,j},$$

where the values $\{\hat{c}_{i,j}\}$ are the coefficients found from the subproblem and $\{c_{i,j}^{(r)}\}$ are the coefficients of the overall approximation after r subproblems. This approximation is evaluated at each data point, and a set of *modified function values* (or residuals) are formed as

$$\mathbf{f}^{(r)} = \mathbf{e}^{(r)} = \mathbf{f}^{(r-1)} - \mathbf{s}^{(r)} = \mathbf{f}^{(r-1)} - A\mathbf{c}^{(r)}$$

where $\mathbf{s}^{(r)}$ is the vector of spline values at the data abscissae. Taking these residuals to be the data ordinates for the subsequent approximation problem means that the overall approximation will approximate the correct data values.

The serial algorithm is as follows.

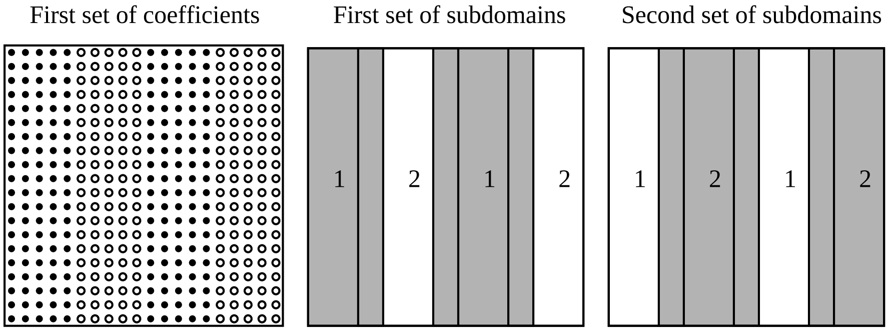
- Choose a small rectangular subregion that supports approximately $3n_x \times 3n_y$ coefficients and calculate the fit on that region.
- Place the coefficients in the correct position in the overall two-dimensional coefficient array.
- Calculate the modified function values at the data points.
- Update the data values by replacing them with the modified function values.
- Iterate this procedure until the residuals meet some convergence criterion.

The subregions are chosen by analysing the residuals in the domain and finding the region where they are largest. Anderson [And94] has proved that this method converges globally to the true least squares approximation to the data, and, for large approximation problems, he suggests that the algorithm is of $\mathcal{O}(mn_y^2)$ [And97]. This makes it faster by $\mathcal{O}(q_x^2)$ when compared to the standard global solution methods.

4 Parallel Algorithm in Two Dimensions

The serial algorithm described above has the potential to work in a parallel environment. Instead of approximating on just one subregion, we can fit on two or more of these regions simultaneously, the only restriction on these regions being that they do not overlap. If this restriction is violated, we would be finding two approximations to some of the data values. This introduces overfitting into the approximation, the problem that the serial algorithm had without modified function values. Choosing enough distinct subregions to employ all of the processing resources is relatively easy for small numbers of processors. However as the number of processors increases with respect to the total number of coefficients to be found, it becomes increasingly

Figure 1 Selecting subdomains by splitting in only one dimension (strips).



difficult to define dynamically sufficient distinct subregions. Also, in a distributed memory environment, choosing subsets of coefficients dynamically will necessitate a lot of unwanted data transfer between processors. To overcome all of these problems we require some systematic way of splitting the domain so that the subdomains are predefined and fixed rather than chosen dynamically.

Recall that the floating point operation count in the solution of the two-dimensional problem is $\mathcal{O}(mq_x^2n_y^2)$. The small subdomains used by the serial algorithm, have smaller values of q_x and, as a result, require far fewer operations to solve. However the number of coefficients in only one of the two dimensions affects the speed of the solution of the system. Therefore subdividing the coefficient set, and the domain, in the second dimension will not affect the global time taken to find the overall approximation.

For the parallel algorithm we consider subdividing the coefficient set in just one dimension, which splits the domain into long thin overlapping strips (see Figure 1). With this decomposition of the domain, we see from the second diagram in Figure 1, that each subdomain overlaps with only two neighbouring regions. Therefore, by grouping the subdomains into two sets, containing alternate subdomains, half of the domain can be fitted in parallel. The modified function values have to be calculated after each parallel section to ensure that we avoid overfitting any of the points. After both of the sets of subdomains have been fitted, we have completed one iteration of the parallel algorithm and can begin fitting on the first group of subregions again. Iterating in this way means that the mathematics of the parallel approximation method are the same as the serial method, and therefore the proof of convergence for the serial method [And97] also holds for the parallel method.

Improving Convergence Rate

A true spline fit of order n needs n knots exterior to the data at each end of the data set. For the decomposition methods described here the subdomains that are formed do not have this property. Therefore towards the edges of subdomains, where fewer than n basis functions cover the data points, the approximation is poor. In the serial algorithm the regions of poor fit are not fixed, because of the dynamic way in which the subdomains are chosen, and therefore are not a problem with regards to the convergence of the approximation. However, the parallel algorithm keeps the domains fixed, and so the approximation remains poor in the overlap regions. Hence the approximation takes a comparatively long time to converge. Changing the subdomains at each iteration of the parallel method, in a similar way to the serial algorithm, causes a large amount of data movement and is thus undesirable. An alternative is to ensure that a true spline fit is formed at every point in the domain in each iteration, thus preventing these regions of poor approximation from arising. We achieve this by increasing the amount of overlap in the subregions from $n-1$ knots to $2(n-1)$. This means that $n-1$ coefficients at the extremes of each subdomain are calculated in the neighbouring subdomain as well. This does increase the parallel workload in each iteration by a small amount [Pin97a], but it does not increase the parallel overheads, and the method converges much more rapidly.

Smaller Domains

The only problem with subdividing the domain into strips is that, for smaller data fitting problems using many processors, there may not be sufficient subdomains to employ all of the available processing resources.

For a parallel system with p processors, the length of the *full* knot set in the leading dimension is $K = q_x + n_x$. The minimum value of K, that allows the domain to be split only in that dimension and still employ all of the processors, is given by

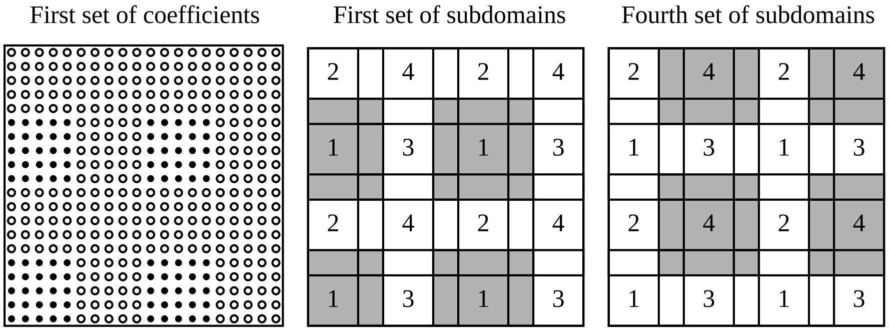
$$K = (4p+2)(n_x - 1).$$

The reasons for this are discussed in [Pin97b]. The most commonly used two-dimensional B-spline fit is bi-cubic ($n_x = n_y = 4$), and the minimum length of the knot set for this order of approximation is thus $6(2p+1)$. If the knot set in the second dimension is of comparable size, then the total number of coefficients needs to approach 40000 before we are able to utilise 16 processors with this decomposition of the domain. Two dimensional approximation problems far smaller in size than this are prohibitively large to solve on a serial machine.

To cope with these smaller domains in a parallel context we must consider subdividing the coefficient array into small rectangular subsets (first diagram in Figure 2). We then take the subdomains to be the regions of support of the sets of B-spline basis functions corresponding to the sets of coefficients (second diagram in Figure 2). Because each region now has as many as eight other subdomains overlapping it, the best that we can do with this decomposition of the domain is to approximate on the domain in four parallel sections. Therefore, more work is introduced into the program, because the modified function values need to be calculated more often, and also there are more synchronisation points in the program because of the increase in distinct parallel sections. However, to effectively utilise 16 processors with this decomposition of the domain we need at least 64 subregions, and on an approximately square domain this can be done by decomposing the domain into a grid of 8×8 subdomains. For a bi-cubic B-spline approximation we need only a minimum of 54×54 (about 3000) coefficients, much less than was the case with the first decomposition method.

On larger domains, however, subdividing the domain in only one dimension and therefore using strips as subdomains is the better choice. This decomposition should allow better load balancing, and less time will be spent calculating the modified

Figure 2 Selecting subdomains by splitting in both dimensions (boxes).


First set of coefficients

First set of subdomains

Fourth set of subdomains

function values.

Decomposition Algorithm

When decomposing the domain in only one dimension, we must ensure that we produce a multiple of $2p$ strips to allow an equal distribution of subdomains across the processors. The width of the subdomain, q_x, must not be too large because of the increase in time taken to solve large domains [And94, Pin97b]. If the problem is not large enough to split in only one dimension, we need to subdivide both dimensions of the domain, and this complicates the decomposition. The decomposition algorithm is summarized below, but a more in depth discussion of it is given in [Pin97b].

1. If $q_x \geq (4p+2)(n_x-1)$, then we can split the domain in one dimension.
   (a) Find s such that

$$(4sp+2)(n_x-1) \leq q_x < (4p(s+1)+2)(n_x-1) \qquad s \in \mathbb{N},$$

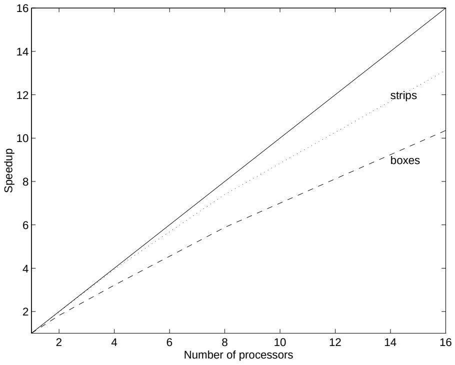
   and split the domain into $2sp$, approximately equal in size, subdomains.

2. Otherwise:
   (a) Find the smallest non-identity factor of p, r say.
   (b) If $q_x \geq (4(p/r)+2)(n_x-1)$ then we may proceed. If this is not the case we use the next smallest factor of p and try the same test until it is satisfied for some r a factor of p.
   (c) Check to ensure that $q_y \geq (4r+2)(n_y-1)$. If this is not the case then we cannot split the domain into enough subregions to employ all of the processors.
   (d) Split the x dimension into $2(p/r)$ equal subdomains and the y dimension into $2r$ equal subdomains to produce a set of $4p$ subdomains.

Implementation on Parallel Architectures

The two-dimensional algorithm is readily implemented on a shared memory architecture such as the KSR-1. Its implementation on a distributed memory - message passing environment is more complicated and the details of this are given in [Pin97b].

Figure 3 Speedup gained on the KSR-1.



5 Results

We ran the program with two different sets of data and knots. The data sets used contained 50,000 and 200,000 points, and these sets were approximated using 54×54 and 204×54 coefficients respectively. Timings for the computational part of the program were made with the data sets on up to 16 processors and these timings can be found in the technical report [Pin97b]. From these timings, the speedups of the parallel system were calculated and these are shown in Figure 3.

The domain with 54×54 coefficients, labelled “boxes” in Figure 3, was decomposed in both dimensions, but the larger problem was decomposed using strips as subdomains and is labelled “strips” in Figure 3. Both methods give very good speedups on the KSR-1. On 16 processors the small problem has a speedup of 10.4 and the larger problem a speedup of 13.1. This difference is due to the better load balancing which is achieved when using strips as subdomains. With optimum load balancing there should in fact be little difference in the speedups obtained from the two decompositions when implemented on the KSR-1.

6 Conclusions

A parallel algorithm that utilises domain decomposition has been described. The results from a practical implementation of the method on the KSR-1 parallel machine show that good speedups can be achieved on two-dimensional problems with data sets which have a uniform distribution of points. The method is sufficiently versatile to be applicable to problems of higher dimension.

Acknowledgement

We should like to thank NAg Ltd. for their support of this work and we should also like to thank Manchester Computing Centre for granting the use of the KSR-1 machine and the London School of Dentistry for supplying the data used in this report.

REFERENCES

[And94] Anderson I. J. (1994) *Efficient multivariate approximation for large sets of structured data.* PhD thesis, University of Huddersfield, Huddersfield, U.K.

[And97] Anderson I. J. (1997) A piecewise approach to piecewise approximation. Preprint.

[Cox82] Cox M. G. (1982) Direct versus iterative methods of solution for multivariate spline-fitting problems. *IMA J. Numer. Anal.* 2: 73–81.

[Cox87] Cox M. G. (1987) Data approximation by splines in one and two variables. In *The State of the Art in Numerical Analysis*, pages 111–138.

[GV89] Golub G. H. and Van Loan C. F. (1989) *Matrix Computations.* John Hopkins University Press, Baltimore, Maryland.

[HH74] Halliday J. and Hayes J. G. (1974) Least squares fitting of cubic spline surfaces to general data. *J. Inst. Math. and Applics.* 14: 89–103.

[Ken93] Kendall Square Research Corporation, Waltham, Ma, USA (October 1993) *KSR Parallel Programming*, second edition.

[Pap93] Papadimitriou P. (December 1993) The KSR-1—A numerical analyst's perspective. Numerical Analysis Report 242, University of Manchester/UMIST.

[Pin97a] Pink C. K. (February 1997) Evaluating B-spline approximations in parallel in one and two dimensions. Technical Report 9705, University of Huddersfield.

[Pin97b] Pink C. K. (January 1997) Parallel B-spline approximation of large sets of uniform data. Technical Report 9706, University of Huddersfield.

Reuse of Krylov Spaces in the Solution of Large-scale Nonlinear Elasticity Problems

Christian Rey and Françoise Léné

1 Introduction

We present the Generalized Krylov Correction, an acceleration technique for the solution to a series of symmetric linear problems with several right-hand sides and matrices, and with its efficiency on an industrial three-dimensional nonlinear elasticity problem. Such a technique is based upon the reuse of Krylov spaces.

Nonlinear elasticity problems are a category of problems often encountered in the field of solid and structural mechanics. The techniques the most generally used for their solution are Newton-type methods [Kel83]. These mainly consist in the construction of a series of linear problems, the solutions of which converge towards the solution to the considered problem. Note that the parent matrices of those linear problems are symmetric positive definite for the type of mechanical problems we consider [Rey94]. The use of non-overlapping domain decomposition methods (primal [LeT94] or dual [FR94] approach) coupled with a conjugate gradient algorithm provides a particularly well-adapted solution for parallel computation for the solution to those symmetric linear problems. However, the numerical efficiency of the conjugate gradient algorithm depends upon the construction of Krylov spaces of the solution to the associated problems. In order to significantly speed up the resolutions of the succession of symmetric linear problems, we developed a technique, well-adapted to an implementation on parallel computers, known as the Generalized Krylov Correction [RDL95] [Rey96]. This method relies upon an efficient utilisation of descent directions calculated in the course of the resolution of previous systems so as to correct the new descent directions. Besides, it can be interpreted as a generalization of the iterative solution of symmetric systems with multiple right-hand sides but with an invariant matrix previously addressed in [Par80], [Saa93] and [FCR94]. This technique is tested on an industrial three-dimensional example (see Fig. 1), a steel-elastomer structure.

Such structures are increasingly used and developed in industry to produce efficient elastic supports. But their numerical computation using the finite element method

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

implies a number of difficulties linked to the great heterogeneousness of the structure, and to the nonlinear behavior of rubber layers. We demonstrate, using this ill-conditioned three-dimensional example, the efficiency of the Generalized Krylov Correction from a numerical point of view and from the point of view of its implementation on parallel computers. We also outline the validity domain of this correction vis-à-vis the evolution of matrices of the various symmetric linear systems. Then, after briefly recollecting of the solution to nonlinear elasticity problems with a Newton-type method, we describe the Generalized Krylov Correction. In the last section, we study the results obtained for the elastic support.

2 Solution to Nonlinear Elasticity Problems

In order to model bodies made up of compressible or quasi-incompressible hyperelastic materials and undergoing large deformations, we choose the Lagrangian formulation. All variables are thus defined and retained in a reference configuration. The equilibrium equations may be written in a weak form as follows :

$$\text{Find } u \text{ such that } \int_{\Omega} \frac{\partial \Phi}{\partial F}(u) : \nabla v \, d\Omega = \int_{\Omega} f.v \, d\Omega + \int_{\partial\Omega} g.v \, d\Gamma \qquad \forall v \tag{2.1}$$

where, $v(x)$, is any admissible displacement field in the reference configuration, $u(x)$, is the displacement field, $f(x)$ and $g(x)$ are the density of body forces and surface tractions respectively, $F(x) = Id + \nabla u(x)$, is the deformation gradient, Φ, is the specific internal elastic energy, and (:), stands for the double-contractor operator.

The problem (2.1) is usually discretized through a finite element method [Cia78] and leads to the solution to a nonlinear problem in the form $G(u) = 0$. Methods classically used for the solution to such problems are Newton-type methods. The simplest method, Newton-Raphson, consists in iteratively replacing in the equation $G(u) = 0$ the function G with its first-order expansion around the point (u). It then iteratively solves the series of linear problems thus obtained.

A first variant of this method (Quasi Newton) consists of reactualizing the linear system matrix only every p nonlinear iterations. However, depending on the stiffness of the problem to be solved, we will use stronger variants, such as that of the Newton Incremental (which consists in incrementing the loading) or even better the so-called bordering algorithm [Kel83]. But whatever the chosen method may be, all the methods come down to the solution to a succession of linear problems, the right-hand sides and matrices of which are to be reactualized.

Refer to Ciarlet [Cia86] for a complete presentation of the formulation of nonlinear elasticity problems and to Keller [Kel83] or Le Tallec [LeT90] for Newton-type nonlinear solvers.

3 Solution to a Succession of Linear Problems

The use of direct solvers for the resolution of such a succession of symmetric linear problems requires huge memory space and is extremely time-consuming when carrying out calculations, particularly for large-scale three-dimensional elasticity problems. Moreover they are not suited for an implementation on parallel computers. This is all the more the case as the matrices of the various problems are to be reactualized.

Domain decomposition methods (primal [LeT94] or dual approach [FR94])), coupled with a conjugate gradient algorithm, make it possible, by condensing problems on the interface of subdomains, to solve the following succession of condensed symmetric linear systems, the parent matrices of which are symmetric, positive definite.

$$C^k \lambda^k = b^k \quad \text{for } k = 1, 2, ..., N \tag{3.2}$$

where C^k is the matrix of Schur complement in dual or primal form according to the approach chosen and b^k the associated condensed right-hand side.

Principle and Initialisation of the Conjugate Gradient

The resolution of the k^{th} of these symmetric linear systems by the algorithm of the conjugate gradient, preconditioned by the matrix M^k generates the following $K(C^k)$ Krylov space, constructed in the course of iterations using descent directions w_i^k [TL87]:

$$K(C^k) = \text{Vect}(w_1^k, ..., w_p^k). \tag{3.3}$$

The λ_p^k p-rank approximation of the solution that minimizes $g = C^k\lambda_p^k - b^k$ residual over the $\{\lambda_0^k\} + K(C^k)$ space (where λ_0^k is a given initial field) for the dot product associated with the C^k matrix is then given by :

$$\lambda_p^k = \lambda_0^k + P(C^k)(b^k - C^k\lambda_0^k) \quad \text{with} \quad P(C^k)(\lambda) = \sum_{i=1}^{p} \frac{(\lambda,\, w_i^k)}{(C^k w_i^k,\, w_i^k)} w_i^k \tag{3.4}$$

The approximation may then define a correct initialization for the resolution of a linear system with the same matrix (the initialization is optimal provided that it is used to restart the resolution of the same linear problem). Moreover, these condensed systems are small in comparison to the dimensions of the global problem; we can therefore keep the information obtained following their resolution using the conjugate gradient method; this approach does not however entail high additional costs (in terms of memory capacity). We may therefore use them in order to speed up the resolution of the following systems.

The Generalized Krylov Correction

A good preconditioner of the k^{th} linear system is the matrix M such that $C^k M g = g$. We propose a g^* approximation [Rey96] of the g^{opt} optimal correction term defined

by the relation, $C^k(M^k g + g^{opt}) = g$, being the orthogonal projection (3.4) associated with the C^{k-1} matrix of the g^{app} solution to the system, $C^{k-1}g^{app} = g - C^{k-1}M^k g$, in the $K(C^{k-1})$ p-dimension Krylov space.
According to (3.4), the g^* correction term may be written :

$$g^* = P(C^{k-1})(g - C^{k-1}M^k g) = P(C^{k-1})(g) - \sum_{i=1}^{p} \frac{(M^k g,\, C^{k-1}w_i^{k-1})}{(C^{k-1}w_i^{k-1},\, w_i^{k-1})} w_i^{k-1} \tag{3.5}$$

This expression underlines that the calculation of the correction term does not require the computation of the $C^{k-1}(M^k g)$ matrix-vector product. The calculation time of this correction therefore has a cost comparable to that of a complete reorthogonalization procedure and consequently remains limited.

However, this correction only requires one Krylov space. So as to extend this first Krylov correction to the previous $k - 1$ Krylov spaces, we proceed by successive corrections. On each iteration of the conjugate gradient, the correction term is calculated as follows :

```
Initialization
        State $z = M^k$ g
        For $i = 1$, to $k - 1$ do
            Compute the correction $g^* = P(C^i)(g - C^i z)$
            State $z = z + g^*$
        End of Loop
Compute the new descent direction $w = z - \frac{(z, C^k w)}{(C^k w, w)} w$
```

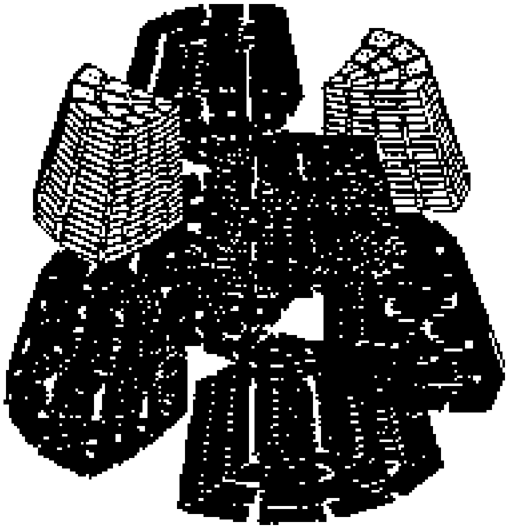
We finally associate to the Krylov correction a complete reorthogonalization procedure [Rou91] of descent directions so as to ensure the correct convergence of the conjugate gradient method.

4 Application

So as to evaluate the efficiency and the validity domain of this technique, we apply it to the resolution of an industrial problem : a steel-elastomer laminated structure (see Fig. 1) subject to an axial compression loading with an imposed displacement. Material behaviors are modelled by the Ciarlet-Geymonat [CG82] specific internal energy Φ. See [Rey94] for a full presentation of the industrial problem.

The problem thus considered is highly nonlinear, and discretized by hexahedral finite elements (Q1 element). The mesh consists of a total of 6435 degrees of freedom and is decomposed into 8 sub-domains (Fig. 1). The domain decomposition method, known as the Dual Schur Complement Method [FR94] coupled with the conjugate gradient algorithm and preconditioned by local rigidity matrices is chosen for the resolution of linear systems. The stopping criterion of this iterative solver is 10^{-3} whereas the stopping criterion of nonlinear iterations (Newton iterations) is 10^{-6}. The difference is justified by the quadratic convergence of the Newton method and by the fact that the solution to linear problems only consists in an intermediate step in the resolution

Figure 1 Decomposition into 8 sub-domains of an elastic support



of the overall problem.

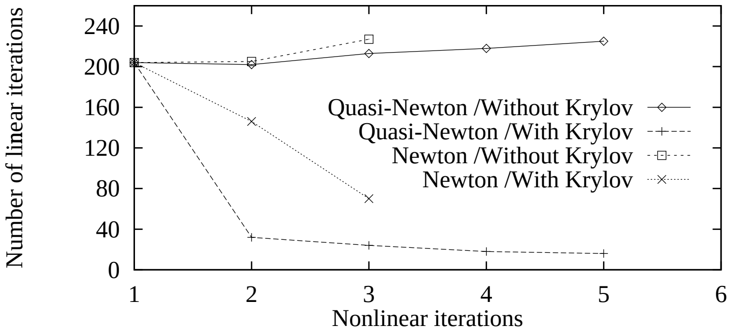
Case where Matrices are not Reactualized

The first case studied is when the nonlinear problem is solved using the Quasi-Newton method without any matrix reactualization. The algorithm then converges in 5 iterations. The number of iterations of the conjugate gradient on each iteration of the Quasi-Newton with or without the addition of the Generalized Krylov correction term (and the initialization introduced in (3.4)) is described in Fig. 2 (Quasi-Newton / Without or With Krylov). It may be observed that introducing the Krylov correction term coupled with the initialization implies a highly significant decrease in the number of iterations. It may thus be observed that this iterative approach corresponds to a semi-direct method which requires only a projection of the solution in Krylov spaces, with a very limited (or even nil) number of the conjugate gradient iterations.

Case of Reactualized Matrices

The nonlinear problem is now solved by the Newton-Raphson method with systematic reactualization of the matrix. The algorithm then converges in 3 iterations. The results obtained with or without the addition of the Generalized Krylov correction term are shown in Fig. 2 (Newton / Without or With Krylov). The decrease in the number of iterations however is smaller than in the previous case but remains significant.

Figure 2 Newton with or Without Generalized Krylov Correction

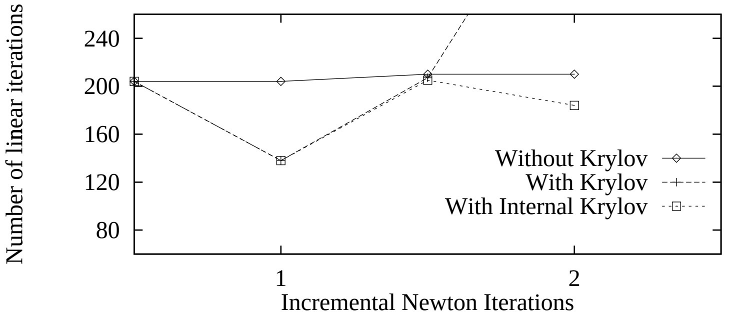


Case of the Incremental Newton Method

In the present section, the problem is solved by the Incremental Newton for a partitioning of the loading into two equal increments. The solution is thus reached after 2 Newton-Raphson iterations for each of the two considered increments. Fig. 3 describes the results obtained in the three following cases :

- without the Generalized Krylov Correction,
- with the Generalized Krylov Correction for the solution of each linear system,
- with the Generalized Krylov Correction only within an Incremental Newton iteration (Internal Krylov).

Figure 3 Incremental Newton with or Without Generalized Krylov Correction



It should be noted that, in the case of the Incremental Newton method, the Krylov Correction fails when it is used for the resolution of each linear system. Indeed, the taking account of the new load increment (here with imposed displacement) implies a very significant evolution of the linear system matrix to be solved and in particularly its spectrum. This explains the loss of efficiency of the Krylov Correction which relies upon the construction of an approximation of the inverse of the linear system matrix. On the other hand, a significant gain may be observed again if the technique is used only for the resolution of nonlinear problems of each load increment (Internal Krylov).

However, its use may be automated when introducing a criterion, which, in the case of a too slow convergence, starts the iteration again without the correction.

5 Conclusion

The introduction of the Generalized Krylov Correction in Domain Decomposition Methods for the resolution of nonlinear elasticity problems may lead to a significant reduction in the number of iterations of the conjugate Gradient. Furthermore, as shown in its expression (3.5), the calculation procedure of the Krylov Correction associated to a Krylov sub-space can be compared to that of a reorthogonalization. It is therefore well suited to an implementation on parallel computers.

REFERENCES

[CG82] Ciarlet P. G. and Geymonat G. (1982) Sur les lois de comportement en élasticité non linéaire compressible. *C.R. Acad. Sci. Paris,* T. 295, Série II: pp. 423–426.

[Cia78] Ciarlet P. G. (1978) *The Finite Element Method for Elliptic Problems.* North-Holland.

[Cia86] Ciarlet P. G. (1986) *Elasticité Tridimensionnelle.* Masson.

[FCR94] Farhat C., Crivelli L., and Roux F. X. (1994) Extending substructure based iterative solvers to multiple load and repeated analyses. *Comput. Meths. Appl. Mech. Engrg.,* 117: pp. 195–209.

[FR94] Farhat C. and Roux F. X. (1994) *Implicit parallel processing in structural mechanics*, volume 2 of *Computational Mechanics Advances.* North-Holland.

[Kel83] Keller H. B. (1983) The bordering algorithm and path following near singular points of higher nullity. *SIAM J. Sci. Sta. Comp.,* 4: pp. 573–582.

[LeT90] LeTallec P. (1990) *Numerical Analysis of Equilibrium Problems in Finite Elasticity.* Université Paris-Dauphine.

[LeT94] LeTallec P. (1994) *Domain decomposition methods in computational mechanics*, volume 1 of *Computational Mechanics Advances.* North-Holland.

[Par80] Parlett B. N. (1980) A new look at the lanczos algorithm for solving symmetric systems of linear equations. *Lin. Alg. Applics.,* 20: pp. 323–346.

[RDL95] Rey C., Devries F., and Léné F. (1995) Parallelism in nonlinear computation of heterogeneous structures. *Calculateurs Parallèles,* 7: pp. 287–316.

[Rey94] Rey C. (Décember 1994) *Développement d'algorithmes parallèles de résolution en calcul non-linéaire de structures hétérogènes : application au cas d'une butée acier-élastomère.* Phd thesis, Ecole Normale Supérieure de Cachan.

[Rey96] Rey C. (1996) Une technique d'accélération de la résolution de problèmes d'élasticité non linéaire par décomposition de domaines. *C.R.Acad. Sci. Paris,* T. 322, Série II b: pp. 601–606.

[Rou91] Roux F. X. (1991) Acceleration of the outer conjugate gradient by reorthogonalisation for a domain-decomposition method for structural analysis problems. *Proceedings of the third international symposium on domain decomposition methods, Houston, Texas, SIAM, Philadelphia* .

[Saa87] Saad Y. (1987) On the lanczos method for solving symmetric linear systems with several right-hand sides. *Math. Comp.,* 48: pp. 651–662.

[TL87] Theodor R. and Lascaux P. (1987) *Analyse numérique matricielle appliquée à l'art de l'ingénieur.* Masson.

56

Preconditioning the FETI Method for Problems with Intra- and Inter-Subdomain Coefficient Jumps

Daniel Rixen and Charbel Farhat

1 Introduction

The FETI method [FR94, MTF, FCR96] and related Balancing algorithm [Man93, LMV95] are two domain decomposition (DD) based iterative solvers that have gained popularity in the last few years. When applied to the solution of problems where the subdomains (a) do not feature neither inter nor intra coefficient jumps, and (b) have good and/or comparable aspect ratios, these DD methods are scalable and quasi-optimal. In order to extend the range of applications where these solvers excel, a simple scaling procedure was described in [LeT94] to address the issue of inter-subdomain coefficient jumps, and a mesh partitioning optimizer was proposed in [FMB95] to remedy the subdomain aspect ratio problem. In this paper, we revisit both issues and present a preconditioning algorithm that addresses the problems of arbitrary subdomain aspect ratios, and large inter- *as well as* intra-subdomain coefficient jumps (so far, most authors have addressed only the problem of inter-subdomain coefficient jumps [LeT94]). The proposed preconditioner is derived from sound energy principles that were initially introduced in [RF96] for improving the accuracy of the solution of subdomain problems by polynomial and piece-wise polynomial Lagrange multipliers. It can be equally used with the FETI and Balanced algorithms. However, because of space limitation, we limit our presentation to the case of the FETI method. We do not offer a mathematical proof of the optimality of our preconditioner, but we demonstrate numerically its scalability with the solution of highly heterogeneous structural mechanics problems.

2 The Focus Problem

The solution of a problem of the form $Ku = f$, where K is a symmetric positive definite matrix arising from the discretization of some second- or fourth-order elliptic

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

problem on a domain Ω, can be obtained by partitioning Ω into N_s subdomains $\Omega^{(s)}$, and gluing these with discrete Lagrange multipliers λ:

$$K^{(s)}u^{(s)} = f^{(s)} - B^{(s)^T}\lambda \qquad s = 1, ..., N_s \tag{2.1}$$

$$\sum_{s=1}^{s=N_s} B^{(s)}u^{(s)} = 0 \tag{2.2}$$

Here, $B^{(s)}$ is a signed subdomain Boolean matrix that extracts and signs the interface components of a vector or a matrix related to $\Omega^{(s)}$. Eliminating $u^{(s)}$ from Eqs. (2.1–2.2) leads to the so-called dual interface problem

$$\begin{bmatrix} F_I & -G \\ -G^T & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \alpha \end{bmatrix} = \begin{bmatrix} d \\ c \end{bmatrix} \tag{2.3}$$

$$F_I = \sum_{s=1}^{s=N_s} B^{(s)}K^{(s)^+}B^{(s)^T}; \quad G = \begin{bmatrix} B^{(1)}R^{(1)} & ... & B^{(N_f)}R^{(N_f)} \end{bmatrix}$$

$$d = \sum_{s=1}^{s=N_s} B^{(s)}K^{(s)^+}f^{(s)}; \quad c = -\begin{bmatrix} f^{(1)^T}R^{(1)} & ... & f^{(N_s)^T}R^{(N_s)} \end{bmatrix}^T$$

where $K^{(s)^+}$ denotes the inverse of $K^{(s)}$ if $\Omega^{(s)}$ is not a floating subdomain, or a generalized inverse if $K^{(s)}$ is singular. In the latter case, $R^{(s)} = Ker(K^{(s)})$ (rigid body modes in structural mechanics), $\alpha = [\alpha^{(1)}...\alpha^{(N_f)}]^T$ where N_f denotes the total number of floating subdomains, and $\alpha^{(s)}$ stores the amplitude coefficients of $R^{(s)}$.

The FETI method consists in constructing the dual interface problem (2.3) and solving this interface problem by a preconditioned conjugate *projected* gradient (PCPG) algorithm where the projector is set to $P = I - G\ (G^TG)^{-1}G^T$. If W is a diagonal matrix which stores for each interface unknown the number of subdomains it belongs to, and $\overline{F_I^{-1}}$ denotes the chosen preconditioner, the FETI algorithm for second-order elasticity problems can be written as summarized in Table 1. (see [FCR96] for an extension to fourth-order elasticity and shell problems).

Two preconditioners have been previously developed for the FETI method: a mathematically optimal Dirichlet preconditioner $\overline{F}_I^{D^{-1}}$, and a computationally economical "lumped" preconditioner $\overline{F}_I^{L^{-1}}$

$$\overline{F}_I^{D^{-1}} = \sum_{s=1}^{s=N_s} B^{(s)} \begin{bmatrix} 0 & 0 \\ 0 & S_{bb}^{(s)} \end{bmatrix} B^{(s)^T} \qquad \overline{F}_I^{L^{-1}} = \sum_{s=1}^{s=N_s} B^{(s)} \begin{bmatrix} 0 & 0 \\ 0 & K_{bb}^{(s)} \end{bmatrix} B^{(s)^T} \tag{2.4}$$

Here, $S_{bb}^{(s)}$ denotes the primal Schur complement associated with subdomain $\Omega^{(s)}$, and the subscripts i and b designate the interior and interface boundary unknowns, respectively.

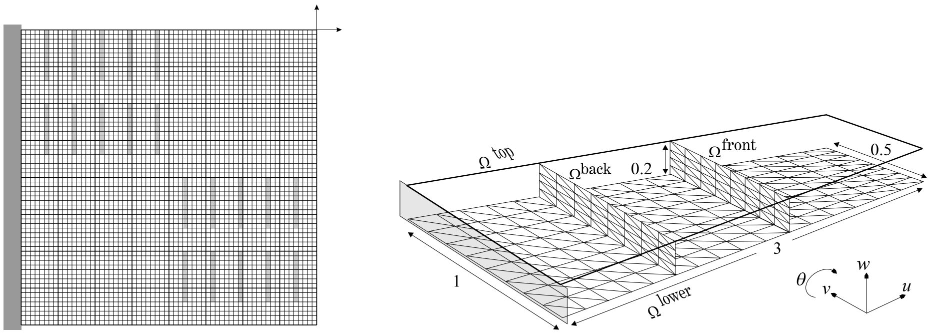
It is well known that the performance of many DD methods including FETI can deteriorate when either material or geometrical heterogeneities are present in the

Table 1 The FETI PCPG method

1. Initialize
$$\lambda^0 = -G\ (G^TG)^{-1}c\ ,\quad r^0 = d - F_I\lambda^0$$

2. Iterate $k = 1,\ 2,\ ...$ until convergence
$$\begin{aligned}
\textit{Project} - \textit{Scale}\quad w^{k-1} &= W^{-1}P^T\ r^{k-1}\\
\textit{Precondition}\quad z^{k-1} &= \overline{F_I^{-1}}w^{k-1}\\
\textit{Re} - \textit{scale} - \textit{Project}\quad y^{k-1} &= W^{-1}P\ z^{k-1}\\
\zeta^k &= {y^{k-1}}^T w^{k-1}/{y^{k-2}}^T w^{k-2}\quad (\zeta^1 = 0)\\
p^k &= y^{k-1} + \zeta^k p^{k-1}\quad (p^1 = y^0)\\
\nu^k &= {y^{k-1}}^T w^{k-1}/{p^k}^T F_I p^k\\
\lambda^k &= \lambda^{k-1} + \nu^k p^k\ ,\quad r^k = r^{k-1} - \nu^k F_I p^k
\end{aligned}$$

Figure 1 Two examples of heterogeneous structures



(a) (b)

vicinity of the subdomain interfaces. Two examples of such problems in structural mechanics are depicted in Fig. 1: (a) a 2D clamped structure featuring inserts of a material that is 1000 times softer than the main material, discretized with 64 × 64 plane stress elements (second-order elasticity) and successively decomposed into 4, 8 and 64 square subdomains; (b) a 3D model of a wing-box structure constructed with DKT plate elements (fourth-order elasticity) and decomposed into subdomains whose interfaces coincide with the intersection of the skin and the stiffeners. Each subdomain with soft inserts in problem (a) is heterogeneous (intra-subdomain heterogeneity) whereas in problem (b) all subdomains are homogeneous, but the mechanical properties are very different depending on the domain orientation (inter-subdomain heterogeneity). In both examples, the subdomain stiffness cannot be characterized by a single coefficient, and therefore the scaling procedure proposed in [LeT94] cannot be applied.

3 Preconditioning with an Energy-based Smoothing Procedure

The Two-subdomain Problem

For the sake of clarity, we consider first the case of a two-subdomain heterogeneous problem. At each iteration of the FETI PCPG algorithm, the matrix-vector product $F_I p^k$ produces a jump across the subdomain interfaces of the iterate solution u^k. In the sequel, we drop the superscript k for simplicity. Elementary mechanics theory suggests that the solution $u^{(s)}$ on the interface boundary of the stiffer subdomain will be closer to the converged solution than the solution on the softer side. This in turn suggests that the computed solution u should be smoothed after each PCPG iteration as follows

$$\begin{aligned} \tilde{u}_b^{(1)} = \tilde{u}_b^{(2)} = \tilde{u}_I &= (1-a)u_b^{(1)} + a u_b^{(2)} && (3.5)\\ \tilde{u}_i^{(s)} &= u_i^{(s)} - K_{ii}^{(s)^{-1}} K_{ib}^{(s)} (\tilde{u}_b^{(s)} - u_b^{(s)}) \qquad s = 1, 2 && (3.6) \end{aligned}$$

which indicates that when the interface solution has been smoothed, a Dirichlet problem must be solved in each subdomain. Of course, the important question is how to select an optimal value of the smoothing parameter a? Let $\delta_I = u_b^{(2)} - u_b^{(1)}$ denote the jump of the solution on the interface Γ_I. After smoothing, the governing equations (2.1) can be written as

$$\begin{bmatrix} K_{ii}^{(1)} & K_{ib}^{(1)} & 0 \\ K_{ib}^{(1)^T} & K_{bb}^{(1)} + K_{bb}^{(2)} & K_{ib}^{(2)} \\ 0 & K_{ib}^{(2)^T} & K_{ii}^{(2)} \end{bmatrix} \begin{bmatrix} \tilde{u}_i^{(1)} \\ \tilde{u}_I \\ \tilde{u}_i^{(2)} \end{bmatrix} = \begin{bmatrix} f_i^{(1)} \\ f_b^{(1)} + f_b^{(2)} \\ f_i^{(2)} \end{bmatrix} + \begin{bmatrix} 0 \\ r_b \\ 0 \end{bmatrix} \qquad (3.7)$$

where r_b is the interface residual induced by smoothing. From (3.5–3.6) and from (2.1), it follows that $r_b(a) = \left(a S_{bb}^{(1)} + (a-1) S_{bb}^{(2)}\right) \delta_I$. Hence, an optimal value of a is one which minimizes r_b. However, rather than minimizing directly some norm of r_b, we propose to adopt a Rayleigh-Ritz approach where the smoothed solutions are viewed as kinematically admissible fields. In view of Eqs. (3.5–3.6–3.7), the total energy can be written as

$$\mathcal{E}(a) = C - 2a\delta_I^T S_{bb}^{(2)} \delta_I + a^2 \delta_I^T (S_{bb}^{(1)} + S_{bb}^{(2)}) \delta_I \qquad (3.8)$$

where C is an expression that does not depend on a. Minimizing $\mathcal{E}(a)$ yields

$$a^D = \frac{k^{(2)^D}}{k^{(1)^D} + k^{(2)^D}}\ , \qquad k^{(1)^D} = \delta_I^T S_{bb}^{(1)} \delta_I\ , \qquad k^{(2)^D} = \delta_I^T S_{bb}^{(2)} \delta_I \qquad (3.9)$$

Here, the superscript D is used to highlight the fact that computing the smoothing parameter a^D requires solving two subdomain Dirichlet problems. Since in general the corrections (3.5–3.6) will create an interface residual $r_b = \Delta f_b^{(1)} + \Delta f_b^{(2)}$ we also propose to correct the Lagrange multipliers iterates as follows

$$\Delta\lambda = -a^D \Delta f_b^{(1)} + (1-a^D)\Delta f_b^{(2)} = -\big(a^D\ S_{bb}^{(1)} a^D + (1-a^D)\ S_{bb}^{(2)}(1-a^D)\big)\delta_I \qquad (3.10)$$

which guarantees the symmetry of our solution method.

¿From a physical viewpoint and in a structural mechanics context, the smoothing procedure proposed here consists in treating two subdomains as two linear springs connected in series, computing the jump of the displacement field at their connection, and redistributing this jump among both springs according to their "relative stiffnesses" $k^{(1)}$ and $k^{(2)}$. While this idea is not new [FR94], the derivation of the smoother yields for the first time a rational estimate of the local measure of a subdomain stiffness.

The Multiple Subdomain Problem - a New Coarse Problem

In order to generalize the smoothing procedure discussed above to the case of an arbitrary number of subdomains, we denote by $b^{(s),j}$ the restriction of the Boolean operator $B^{(s)}$ to the j-th edge of the interface boundary $\Gamma_I^{(s)}$. Using the interior/interface boundary partitioning of the subdomain unknowns we can write

$$B^{(s)} = \begin{bmatrix} 0 & b^{(s),i} & b^{(s),j} & \cdots & b^{(s),l} \end{bmatrix} \tag{3.11}$$

$b^{(s),j}$ can be further decomposed into square submatrices $b^{(sr),j}$ that describe the connectivity of subdomains $\Omega^{(s)}$ and $\Omega^{(r)}$ along edge j. Designating by $r, l \ldots$ the subdomains interconnected with $\Omega^{(s)}$ along Γ_I^j, we have

$$b^{(s),j^T} = \begin{bmatrix} 0 & \cdots & b^{(sr),j^T} & 0 & \cdots & b^{(sl),j^T} & \cdots \end{bmatrix} \tag{3.12}$$

Next, we designate the unsigned equivalents of $b^{(sr),j}$ by a hat, and introduce the operator $\hat{b}^{(sr),j^T} \hat{b}^{(rs),j}$ which gives the correspondence between the numberings of the unknowns on both sides of the interface. Of course, we have $\hat{b}^{(sr),j^T} \hat{b}^{(sr),j} = I$. The generalization to an arbitrary number of subdomains of the smoothing procedure (3.5–3.6) then goes as follows:

$$\begin{aligned} \tilde{u}_b^{(s),j} &= \beta^{(s),j} u_b^{(s),j} + \sum_{r:\Gamma_I^{(r)} \supset \Gamma_I^j}^{r \neq s} \hat{b}^{(sr),j^T} \hat{b}^{(rs),j} \beta^{(r),j} u_b^{(r),j} \qquad \forall \text{ edge } j & (3.13) \\ \tilde{u}_i^{(s)} &= u_i^{(s)} - K_{ii}^{(s)^{-1}} K_{ib}^{(s)} (\tilde{u}_b^{(s)} - u_b^{(s)}) \qquad s = 1, \cdots N_s & (3.14) \end{aligned}$$

where $\beta^{(s),j}$ are scalar smoothing parameters. If the $\beta^{(s),j}$ are constrained to have a unit sum

$$\sum_{\Gamma_I^{(s)} \supset \Gamma_I^j} \beta^{(s),j} = 1 \qquad \forall \text{ edge } j \tag{3.15}$$

the corrections of the subdomain interface solutions can be written as

$$\Delta u_b^{(s),j} = \tilde{u}_b^{(s),j} - u_b^{(s),j} = - \sum_{r:\Gamma_I^{(r)} \supset \Gamma_I^j}^{r \neq s} \beta^{(r),j} \; \hat{b}^{(sr),j^T} \; \delta_I^{(sr),j} \tag{3.16}$$

where $\delta_I^{(sr),j} = \hat{b}^{(sr),j} u_b^{(s),j} - \hat{b}^{(rs),j} u_b^{(r),j}$. To determine the edge coefficients $\beta^{(s),j}$ we follow conceptually the same Rayleigh-Ritz approach as presented in Section 3. If the unit sum condition is enforced by a set of multipliers τ_j, the minimization of the total energy leads to the following *auxiliary coarse problem*

$$\sum_{s:\Gamma_I^{(s)}\supset\Gamma_I^j}^{s\neq q} \quad \sum_{i:\Gamma_I^{(s)}\supset\Gamma_I^i} \quad \sum_{p:\Gamma_I^{(p)}\supset\Gamma_I^i}^{p\neq s} k_s^{(q),j;(p),i} \;\; \beta^{(p),i} \;=\; \tau_j \qquad\qquad \forall[(q),j] : \Gamma_I^{(q)} \ni j \tag{3.17}$$

$$\text{where} \quad k_s^{(q),j;(p),i} = \left(\hat{b}^{(sq),j}\delta_I^{(qs)}\right)^T [S_{bb}^{(s)}]_{j,i} \left(\hat{b}^{(sp),i}\delta_I^{(ps)}\right) \tag{3.18}$$

and $[S_{bb}^{(s)}]_{j,i}$ is the Schur-complement of $K^{(s)}$ associated with the edges j and i.

For symmetry, the correction of the Lagrange multipliers introduced by between $\Omega^{(s)}$ and $\Omega^{(r)}$ along edge j is then computed as

$$\Delta\lambda^{(sr),j} = b^{(sr),j}\beta^{(r),j}\Delta f_b^{(s),j} + b^{(rs),j}\beta^{(s),j}\Delta f_b^{(r),j} \tag{3.19}$$

where $\Delta f_b^{(s)} = S_{bb}^{(s)}(\tilde{u}_b^{(s)} - \tilde{u}_b^{(s)})$. In summary, using the notation of Table 1, this preconditioning step can be written as

$$z^{k-1} = \left\{ \sum_{s=1}^{s=N_s} \beta^{(s)} \; B^{(s)} \begin{bmatrix} 0 & 0 \\ 0 & S_{bb}^{(s)} \end{bmatrix} B^{(s)^T} \; \beta^{(s)^T} \right\} w^{k-1} \tag{3.20}$$

Cost-effective Alternatives — Lumping and "Superlumping"

The smoothing procedure presented in Sections 3 and 3 requires solving in each subdomain several Dirichlet problems. A first economical variant can be designed by replacing $\tilde{u}_i^{(s)} = u_i^{(s)}$ in (3.14), which has the effect of not propagating the correction of interface smoothing to the subdomain interior unknowns. It can be shown that such a strategy leads to similar expressions of the smoothing coefficients, but replaces the expensive Dirichlet operators $S_{bb}^{(s)}$ by the more economical lumping matrices $K_{bb}^{(s)}$ in the expression (3.20) and in the computation of the interface stiffnesses (3.18). This lumped preconditioner does no longer take into account the intra-subdomain heterogeneities associated with internal nodes. Nevertheless, the heterogeneities associated with the elements on the interface are still treated correctly.

Noting that the auxiliary coarse problem must be reconstructed at each iteration, an even more economical variant for computing the smoothing parameters $\beta^{(s)}$ in the lumped preconditioner can be constructed by assuming that the total energy of the system can be "superlumped" and written as

$$\mathcal{E}(\beta^{(s),j}) = C + \frac{1}{2}\sum_{s=1}^{s=N_s} \Delta u_b^{(s)^T} K_{bb,diag}^{(s)} \; \Delta u_b^{(s)} \tag{3.21}$$

where $K_{bb,diag}^{(s)}$ denotes the diagonal part of $K_{bb}^{(s)}$. In that case, the smoothed interface solution is still given by (3.13), but $\beta^{(r),j}$ is now understood as the diagonal matrix of the interface smoothing parameters (one coefficient per unknown). The unit sum constraint is then expressed at the unknown level. Noting $c^{(sr),j^T} = \hat{b}^{(sr),j^T}\hat{b}^{(rs),j}$ the correspondence between interface numberings, the generalization of the unit sum constraint and (3.16) can be written as

$$\beta^{(s),j} + \sum_{r:\Gamma_I^{(r)} \supset \Gamma_I^j}^{r \neq q} c^{(sr),j^T} \beta^{(r),j} c^{(sr),j} = I \qquad \forall \text{ edge } j \tag{3.22}$$

$$\Delta u_b^{(s),j} = - \sum_{r:\Gamma_I^{(r)} \supset \Gamma_I^j}^{r \neq s} \left(c^{(sr),j^T} \beta^{(r),j} c^{(sr),j} \right) \hat{b}^{(sr),j^T} \delta_I^{(sr),j} \tag{3.23}$$

¿From (3.21), it follows that

$$\beta^{(s),j} = \left[K_{bb,diag}^{(s)} \right]_j \left\{ \sum_{r:\Gamma_I^{(r)} \supset \Gamma_I^j} c^{(sr),j^T} \left[K_{bb,diag}^{(r)} \right]_j c^{(sr),j} \right\}^{-1} \tag{3.24}$$

where $c^{(ss),j} = I$. Hence, for this second smoothing alternative referred to here as the superlumped one, the auxiliary coarse problem is diagonal and needs to be constructed only once. Therefore, implementing it is trivial and solving it is inexpensive.

4 Numerical Results

We consider again problems (a) and (b) depicted in Fig 1, and perform their linear static analysis using the FETI method with the Dirichlet and lumped preconditioners, as well as the various smoothing procedures presented in this paper. We report in Table 2 the number of FETI PCPG iterations.

Table 2 Performance results ($||Ku - f||_2 \leq 10^{-6}||f||_2$)

		Nbr. of PCPG iterations					
			Dirichlet			*lumped*	
Decomposition	$N \times M$	–	smooth.	hyper.	–	smooth.	hyper.
	2×2	18	16	17	35	35	36
plane stress	4×4	63	23	26	80	48	47
	8×8	83	27	25	89	46	44
stiffened panel with 4 subdomains		122	115	25	128	116	50

For the 2D plane stress problem (a), the full smoothing method and its superlumped alternative yield very similar convergence rates, and both of them improve dramatically the performances of the Dirichlet and lumped preconditioners. Table 2 also demonstrates the scalability of the overall solution method with respect to the number of subdomains.

For the stiffened panel problem, the full smoothing procedure improves only slightly the convergence of the FETI method, whereas the superlumped variant reduces the number of iterations by a factor of 5 (Dirichlet preconditioner), and by a factor greater than two (lumped preconditioner). For this problem, the poor efficiency of the full smoothing method can be explained by the fact that one coefficient cannot characterize an interface stiffness, because the relative interface stiffnesses at the intersection between the stiffeners and the skin depend strongly on the direction of the displacement unknown.

REFERENCES

[FCR96] Farhat C., Chen P., and Roux F. (1996) The two-level FETI method - part II: Extension to shell problems. parallel implementation and performance results. *Comp. Meths. Appl. Mech. Eng.* in press.

[FMB95] Farhat C., Maman N., and Brown G. (1995) Mesh partitioning for implicit computations via iterative domain decomposition: impact and optimization of the subdomain aspect ratio. *Int. J. Numer. Meths. Eng.* 38: 989–1000.

[FR94] Farhat C. and Roux F. (1994) Implicit parallel processing in structural mechanics. *Comp. Mech. Adv.* 2(1): 1–124.

[LeT94] LeTallec P. (1994) Domain-decomposition methods in computational mechanics. *Comp. Mech. Adv.* 1: 121–220.

[LMV95] LeTallec P., Mandel J., and Vidrascu M. (1995) Balancing domain decomposition for plates. In Keyes D. E. and Xu J. (eds) *Proc. Seventh Int. Conf. on Domain Decomposition Meths.*, number 180 in Contemporary Mathematics, pages 15–24. AMS, Providence.

[Man93] Mandel J. (1993) Balancing domain decomposition. *Comm. Appl. Num. Meth.* 9: 233–241.

[MTF] Mandel J., Tezaur R., and Farhat C.An optimal Lagrange multiplier based domain decomposition method for plate bending problems. *SIAM J. Sc. Stat. Comput.* (submitted).

[RF96] Rixen D. and Farhat C. (April 1996) Highly accurate and stable algorithms for the static and dynamic analyses of independently modeled substructures. In *Structures, Structural Dynamics and Material Conference.* 37rd AIAA/ASME/ASCE/AHS/ASC, Salt Lake City.

[RFG95] Rixen D., Farhat C., and Géradin M. (1995) Approximation du préconditionneur de dirichlet pour la résolution itérative du problème d'interface de la méthode hybride FETI. In Hermes (ed) *Deuxìeme Colloque National en Calcul des Structures, Giens*, volume 2, pages 655–660.

57

Parallel Implementation of the Two-level FETI Method

François-Xavier Roux and Charbel Farhat

1 Introduction

Recently, a new preconditioning technique for the FETI method based upon a coarse grid problem associated with interface crosspoints has been introduced [MTF]. This gives optimal convergence property for high-order problems. In the present paper the problem of the parallel implementation of this new preconditioning technique is addressed and the performance of this approach is demonstrated for real life structural analysis problems.

For fourth-order problems, like plate or shell problems, the singularity with interface crosspoints, that means nodes that belongs to more than two subdomains, deteriorates the condition number of the dual Schur complement operator, the condensed interface operator defining the FETI method [FR94].

A new preconditioning technique leading to a two-level handling of interface continuity requirements has been recently developed [MTF]. The independence upon the number of subdomains and the polylogarithmical dependence upon the number of elements per subdomain of the condition number of the preconditioned interface problem has been proved.

In the present paper the problem of the parallel implementation of this new preconditioning technique is addressed. After recalling the principle of the FETI method, the new preconditioning technique is introduced and reinterpreted as a two-level FETI algorithm. A parallel implementation strategy is derived from this formulation. Last, the performance of this approach is demonstrated for real life structural analysis problems on an Intel-PARAGON system.

2 The FETI Method

The FETI method is based on introducing the Lagrange multiplier of the continuity condition on interfaces between subdomains. In the case of linear elasticity equations, the Lagrange multipler is equal to the field of interaction forces between subdomains.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

In each subdomain, Ω_i, the local displacement field is solution of the linear elasticity equations with external loadings and boundary conditions inherited from the complete problem, and imposed forces (Neumann boundary conditions) on the interfaces with other subdomains.
With a finite element discretization, this leads to the following set of equations:

$$K_i u_i = B_i^t \lambda + b_i \tag{2.1}$$

where K_i is the stiffness matrix, u_i the displacement field, B_i a signed boolean matrix associated with the discrete trace operator, and λ the Lagrange multipler.
The continuity requirement along the interfaces is written as follows:

$$\sum_i B_i u_i = 0 \tag{2.2}$$

where the signed discrete trace matrices B_i are such that if subdomains Ω_i and Ω_j are connected by the interface Γ_{ij}, then restriction of equation (2.2) on Γ_{ij} is: $u_i - u_j = 0$.
In most subdomains, local problems (2.1) are ill posed, because only Neumann boundary conditions are imposed.
So, if K_i^+ is a pseudo-inverse of matrix K_i, and if columns of matrix N_i form a basis of the kernel of K_i (rigid body motions), equation (2.1) is equivalent to:

$$\left\{ \begin{array}{c} u_i = K_i^+[b_i + B_i^t\lambda] + N_i\alpha_i \\ N_i^t[b_i + B_i^t\lambda] = 0 \end{array} \right. \tag{2.3}$$

The first equation means that the solution of the problem is defined as the sum of a particular solution computed using the pseudo-inverse of K_i plus an element of the kernel. The second equation means that the right-hand side of equation (2.1) must be in the image space of K_i.
Introducing u_i given by equation (2.3) in the continuity condition (2.2) gives:

$$\sum_i B_i K_i^+ B_i^t \lambda + \sum_i B_i N_i \alpha_i = -\sum_i B_i K_i^+ b_i \tag{2.4}$$

With the constraint on λ set by the second equation of (2.3), the global interface problem can be written:

$$\begin{bmatrix} D & -G \\ -G^t & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ \alpha \end{bmatrix} = \begin{bmatrix} d \\ c \end{bmatrix} \tag{2.5}$$

With:

- $D = \sum B_i K_i^+ B_i^t$, dual Schur complement matrix,
- $G\alpha = \sum B_i N_i \alpha_i$, jump of rigid body motions defined by α_i in Ω_i,
- $(G^t\lambda)_i = N_i^t B_i^t \lambda$, $d = -\sum B_i K_i^+ b_i$, $c_i = -N_i^t b_i$.

3 Parallel Solution of the Condensed Interface Problem

The number of constraints of the hybrid condensed interface problem (2.5) is the total number of rigid body modes. As this number is low, the projector associated with this constraint can be explicitly computed:

$$P = I - G(G^tG)^{-1}G^t \tag{3.6}$$

The computation of the product by projector P requires products by G and G^t and the solution of systems with form:

$$(G^tG)\alpha = G^tg \tag{3.7}$$

The product by G^t can be performed independently in each subdomain, the product by G requires exchanging data through interfaces between neighbouring subdomains. Both products can be easily performed in parallel in a message passing programming environment.

Parallelizing the solution of problem (3.7) is more challenging, because of its global implicit nature. To avoid the construction of matrix G^tG, this problem can be solved by the conjugate gradient algorithm. Then only products by G and G^t that can be performed in parallel are required.

Applying the projected conjugate gradient algorithm to the condensed interface problem (2.5), requires the following two main steps.

1. Given an approximate value λ^p, compute the particular solution of the local Neumann problem in each subdomain:

$$u_i^{p+} = K_i^+[b_i + B_i^t\lambda^p] \tag{3.8}$$

and compute the jump of the local displacement fields along interfaces between subdomains that is the gradient of the condensed interface problem:

$$g^p = \sum_i B_iu_i^{p+} \tag{3.9}$$

2. Compute the projected gradient, Pg^p given by formula:

$$Pg^p = g^p + G\alpha^p \text{ with } (G^tG)\alpha^p = -G^tg^p \tag{3.10}$$

The projection step consists in fact in computing the rigid body motions coefficients α_i that minimize the jump of the complete displacement fields given by:

$$u_i^p = u_i^{p+} + N_i\alpha_i \tag{3.11}$$

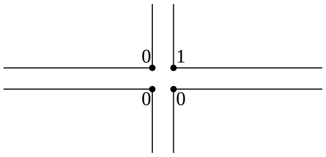
This minimization is performed in the sense that the jump of the complete displacement fields u_i^p, which is in fact the projected gradient, is orthogonal to the traces of all the local rigid body modes:

$$Pg^p = \sum_i B_iu_i^{p+} + \sum_i B_iN_i\alpha_i = \sum_i B_iu_i^p \tag{3.12}$$

$$(G\beta)^tPg^p = 0 \; \forall\beta \;\Leftrightarrow\; (B_iN_i)^tPg^p = 0 \; \forall i \tag{3.13}$$

This projection phase consists in solving a global coarse problem associated with the rigid body coefficients. The condition number of the projected dual Schur complement can be proved to be independent upon the number of subdomains ([FMR94]).

Figure 1 A “corner mode” for a scalar problem



4 The Second-level FETI Preconditioner

With domain decomposition method using local Neumann problems, the jumps of local solution fields at interface crosspoints can be discontinuous. For second-order problems, this singularity entails a polylogarithmic dependence of the condition number upon the local mesh size h/H ([Le 94]). For higher order problems, like plate and shell problems in structural analysis, the singularity is polynomial. So it is of great importance to get rid of these singularities.

The solution consists in constraining the Lagrange multiplier to generate local displacement fields that are continuous at interface crosspoints. Enforcing this constraint induces another level of preconditioning for the FETI method, that restores the optimality property of the FETI method ([FM]).

To get a practical formulation of this constraint, it can be observed that requiring the continuity of displacement fields at interface crosspoints is equivalent to imposing their jump to be orthogonal to the jump of “corner modes” defined as displacement fields with unit value in one space direction at a node connected to a crosspoint as in Figure 1.

Note C_i the set of corner modes in subdomain Ω_i, then the Lagrange multiplier λ^p satisfies the continuity requirement of associated displacement fields at interface crosspoints if the projected gradient satisfies:

$$(B_iC_i)^tPg^p = 0 \ \forall i \ \Leftrightarrow \ (\sum_i B_iC_i\gamma_i)^tPg^p = 0 \ \forall\gamma \tag{4.14}$$

The analogy between constraints defined by equations (3.13) and (4.14) suggests that the preconditioner can be constructed as a correction based upon jumps of corner modes in the same way as the projected gradient is constructed as a correction of the gradient based upon jumps of rigid body modes in (3.12).

$$MPg^p = Pg^p + \sum_i B_iC_i\delta_i \tag{4.15}$$

In term of structural analysis, this means that correcting the interaction forces at interface crosspoints should be enough to make the local displacement fields continuous at these points. In fact, the direction vector must be constructed from the projection of the preconditioned vector MPg^p to satisfy the constraint of orthogonality to the traces of rigid body modes (3.13). This projected preconditioned projected gradient takes the following form:

$$w^p = PMPg^p = Pg^p + \sum_i B_iC_i\delta_i + \sum_i B_iN_i\alpha_i \tag{4.16}$$

The variation of displacement fields induced by the variation w^p of interaction forces must have a null jump at interface crosspoints. By definition of the dual Schur complement, the jump of displacement fields induced by interface forces w^p is PDw^p. According to (4.14) this condition can be written:

$$(\sum_i B_i C_i \gamma_i)^t PDw^p = 0 \ \forall \gamma \tag{4.17}$$

By definition of projector P given by (3.13), this jump satisfies also :

$$(\sum_i B_i N_i \beta_i)^t PDw^p = 0 \ \forall \beta \tag{4.18}$$

Introduce now the coarse subspace defined by both the rigid body modes and the corner modes of all subdomains. Given α_i and δ_i, coefficients of rigid body motions and corner displacements in each subdomain Ω_i, define local coarse grid coefficients ξ_i by merging α_i and δ_i vectors. Then, the global coarse correction of interaction forces is defined as:

$$C_G \xi = (\sum_i B_i N_i \alpha_i) + (\sum_i B_i C_i \delta_i) \tag{4.19}$$

Thus the direction vector w^p takes form: $w^p = Pg^p + C_G \xi$ (4.20)

From equations (4.18) and (4.17) the coarse correction must satisfy the variational equality:

$$(C_G\zeta)^t PDw^p = (C_G\zeta)^t PDPw^p = 0 \ \forall \zeta \ \Leftrightarrow \ (C_G\zeta)^t PDPC_G\xi = -(C_G\zeta)^t Pg^p \ \forall \zeta \tag{4.21}$$

From (4.16), w^p must satisfy the constraint:

$$Pw^p = w^p \ \Leftrightarrow \ PC_G\xi = C_G\xi \tag{4.22}$$

Equations (4.21) and (4.22) represent in fact a constrained variational problem for the coarse grid space defined by rigid body and corner modes, which is similar to problem (2.5). Its formulation as an hybrid algebraic system of equation can be written:

$$\begin{bmatrix} C_G^t D C_G & -C_G^t G \\ -G^t C_G & 0 \end{bmatrix} \begin{bmatrix} \xi \\ \beta \end{bmatrix} = \begin{bmatrix} -C_G^t P g^p \\ 0 \end{bmatrix} \tag{4.23}$$

This system is precisely a coarse FETI problem, posed in the subspace of Lagrange multipliers defined as the image space of C_G. With this coarse grid preconditioner, the solution algorithm appears clearly as a two-level FETI method: at each iteration of projected conjugate gradient at the fine level, an additional preconditioning problem of the same type has to be solved at the coarse grid level.

5 Parallel Implementation and Performance Results

To keep a local representation of each operator, and to exploit domain-based parallelism, both the fine and the coarse grid problems must be solved through the same projected gradient procedure. However, this approach may appear very costly for the coarse grid problem, firstly because this problem must be solved exactly at each projected conjugate gradient iteration for the fine grid problem, and secondly because each coarse grid iteration is as expensive as a fine grid iteration, as local problems are the same for both.

In order to limit the cost, the coarse grid dual Schur complement can be preassembled at each subdomain level. In practice, if ζ_j is a coarse vector that has non zero entries only in subdomain Ω_j, $DC_G\zeta_j$ is non zero only in neighboring subdomains Ω_i. Hence, to precompute the $C_G^tDC_G$ matrix in each subdomain, it is necessary to solve the Neumann problem in subdomain Ω_i for each coarse grid mode in neighbouring subdomain Ω_j with non zero trace on interface Γ_{ij}.

The solution time for solving fine grid problems iteratively can be also drastically reduced using the restarted (projected) conjugate gradient technique presented in [Rou95]. The principle is as follows: if a set of conjugate directions (w^k), $1 \leq k \leq p$, is given, then the element x^0_{start} of x^0+Span$\{w^1, w^2, \dots, w^p\}$ that minimizes the residual can be easily computed:

$$x^0_{start} = x^0 - \sum_{k=1}^{p} \frac{(g^0, w^k)}{(Aw^k, w^k)} w^k \tag{5.24}$$

Applying the standard conjugate gradient algorithm from starting vector x^0_{start} does not ensure that the new direction vectors are conjugate to the vectors w^k. To enforce these additional conjugacy relations, the new direction vector d^q at iteration number q must be reconjugated to the vectors w^k through the following procedure:

$$d^q = g^q - \frac{(g^q, Ad^{q-1})}{(Ad^{q-1}, d^{q-1})} d^{q-1} - \sum_{k=1}^{p} \frac{(g^q, Aw^k)}{(Aw^k, w^k)} w^k \ , \tag{5.25}$$

where g^q is the gradient vector at iteration q, $g^q = Ax^q - b$.

When this procedure is applied for successive right-hand sides with accumulation of conjugate direction vectors, it finally consists in using the conjugate gradient algorithm as a direct solver with an explicit computation of the inverse problem. For the two-level FETI method, this technique is applied for both coarse grid problems. As the dimensions of these problems are small, the procedure is very efficient numerically, and it can be parallelized via a domain-based approach.

Table 1 gives the performance results for a real life application of the two-level FETI method on an Intel-PARAGON machine with increasing number of subdomains and processors. The model is a submarine shell structure featuring 60332 nodes and 362000 degrees of freedom. The first 4 columns of this table give the numbers of subdomains, rigid body modes, corner modes and iterations at fine level. Column 5 features the condition number of the preconditioned condensed interface problem. Columns 6 and 7 give the parallel wall clock times for the initialization phase, including local matrices factorization and coarse matrix forming, and for the iterative solution phase. In both

Table 1 Comparison of parallel performance of one- and two-level FETI

number of					timings	
proc.	rigid body modes	corner modes	iter.	CN	init. (coarse)	total iter. (coarse)
Two-level FETI, local Dirichlet preconditioner						
30	132	351	93	822	361 (108)	514 (74)
40	168	474	94	662	298 (86)	453 (92)
60	318	762	105	828	128 (57)	355 (146)
80	396	954	87	537	69 (34)	240 (128)
One-level FETI, local Dirichlet preconditioner						
30	132	0	289	367398	253	1374
40	168	0	312	3206569	212	1217
60	318	0	406	3505918	71	869
80	396	0	416	315799	35	597

columns the fraction of time spent at coarse grid level is given for the two-level case. According to theory, thanks to the coarse grid preconditioner associated with rigid body modes, the condition number should not depend upon the number of subdomains but only upon a the local mesh size h/H. For a fixed global problem, increasing the number of subdomains increases the local mesh size, and consequently should decrease the condition number. But the condition number depends also upon the aspect ratio of subdomains. In the case of the real life problems like the one presented here, different mesh splittings lead to different aspect ratios of subdomains. Hence, although the local mesh size increases with the number of subdomains, the condition number does not necessarily decrease in a regular way.

Nevertheless, these tests show firstly the great improvement of condition number due to the second level FETI preconditioner. Secondly, it leads to a significant decrease, more than a factor of 2, of the total solution time compared to the one-level method. Thirdly, the parallel implementation exhibits a good scalability, although the time spent for coarse grid iterations reaches 50% of the solution time. These results are very representative of the ones obtained for various industrial problems.

6 Conclusion

The two-level FETI method appears to be a very efficient method for solving real life shell or plate problems. With a parallel implementation using domain-based parallelism and the restarted conjugate gradient method for coarse grids problems, the performance of the method on distributed memory parallel machines with message passing programming environment is already quite satisfactory. Nevertheless, there is still room for improvement in the parallel solution of coarse grid problems.

REFERENCES

[FM] Farhat C. and Mandel J. The two-level FETI method for static and dynamic plate problems- part1: an optimal iterative solver for biharmonic systems. *Comput. Meths. Appl. Mech. Engrg. (submitted)* .

[FMR94] Farhat C., Mandel J., and Roux F.-X. (1994) Optimal convergence properties of the feti domain decomposition method. *Comput. Meths. Appl. Mech. Engrg.* 115: 367–388.

[FR94] Farhat C. and Roux F.-X. (1994) In Oden J. T. (ed) *Implicit parallel processing in structural mechanics*, volume 2 of *Computational Mechanics Advances*, pages 1–124. Nort-Holland.

[Le 94] Le Tallec P. (1994) In Oden J. T. (ed) *Domain decomposition methods in computational mechanics*, volume 1 of *Computational Mechanics Advances*, pages 121–220. Nort-Holland.

[MTF] Mandel J., Tezaur R., and Farhat C. An optimal Lagrange multiplier based domain decomposition method for plate bending problems. *SIAM J. Sc. Stat. Comput. (submitted)* .

[Rou95] Roux F.-X. (1995) Parallel implementation of a domain decomposition method for non-linear elasticity problems. In David E. Keyes Y. S. and Truhlar D. G. (eds) *Domain-Based Parallelism and Problem decomposition Methods in Computational Science and Engineering*, pages 161–176. SIAM, Philadelphia.

58

Implementation of Multigrid on Parallel Machines Using Adaptive Finite Element Methods

Linda Stals

1 Introduction

Multigrid methods and adaptive finite element methods are well established as powerful tools for the solution of partial differential equations. When implementing these methods, parallel computers are often considered because of their ability to solve large problems quickly. However, in practice the implementation of multigrid methods on these machines is non-trivial due to the intra-grid and inter-grid data dependencies. Furthermore, the non-uniformity of the grids generated by adaptive refinement leads to a 'conflict of interest' as parallel machines are better suited to uniform grids. Consequently the implementation of these methods in parallel is a sizeable software engineering problem. In this paper we describe a parallel implementation based upon the use of a node-edge data structure.

The node-edge data structure stores the grids by placing the geometrical information in the node table and the topological information in the edge table. To use the data structure in parallel we have also included a ghost-node table. The ghost-node table is a generalisation of the boundary layer or artificial boundary often used in the parallel implementation of structured grids.

The grids are refined by using the newest node bisection method. We have developed a parallel extension of Mitchell's compatibly divisible triangle method ([Mit88], [Mit89], [Mit92]) to ensure that the angles remain bound away from 0 and π during adaptive refinement.

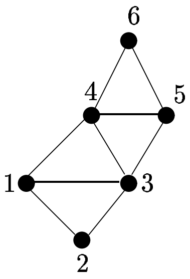
This paper also gives some example runs.

2 Node-Edge Data Structure

The basic idea behind the node-edge data structure is easily illustrated by an example. Consider the grid in figure 1. It can be broken down in terms of its geometrical and

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Figure 1 Example grid stored in the node-edge data structure.



topological components and stored in the following node and edge tables:

Node: 1(0.0, 0.0), 2(3.0, -3.0), 3(5.0, 0.0), 4(3.0, 3.0), 5(7.0, 3.0), 6(5.0, 5.0)

Edge: 1-2, 1-3, 1-4, 2-1, 2-3, 3-1, 3-4, 3-5, 4-1, 4-3, 4-5, 4-6, 5-3, 5-4, 5-6, 6-4, 6-5

Our implementation of this data structure is based upon the one given by Rüde in [Rüd92], [Rüd93a] and [Rüd93b].

The advantage of this data structure is its flexibility. The same data structure can be used to store triangles, quadrilaterals and tetrahedrons. Most of our work has concentrated on triangular grids, but we have developed modules which use the node-edge data structure to store bilinear basis functions and we have started work on tetrahedrons.

The stiffness matrix is stored in a connection table. If the entry in the stiffness matrix corresponding to, say, node i and node j is non-zero then nodes i and j are connected. For linear basis functions, the connection table looks very similar to the edge table. The connection table is also used to store other algebraic information such as the interpolation and restriction operators used in the multigrid algorithm.

3 Parallel Implementation

To use the data structure in parallel we include a ghost-node table and a neighbour-node table. Note that in the parallel implementation we call the node table the full-node table, this helps to differentiate between the whole grid and the grid segments stored in the processors.

The concepts behind the ghost-node table are easily shown by an example. Consider the grid given in Figure 1, and suppose nodes 4, 5 and 6 were placed in the full-node table for one processor while nodes 1, 2 and 3 were placed into the full-node table of another processor. Then the edges for the first processor are completed by adding nodes 1 and 3 as ghost-nodes and the edges for the other processor are completed by adding nodes 4 and 5 as ghost-nodes (see Figure 2).

The example in Figure 2 showed how the ghost-nodes complete the edge table, but they are also used to complete the connection table and consequently complete the

Figure 2 Example grid spread over two processors. The ghost-nodes, shown by open circles, complete the edges.

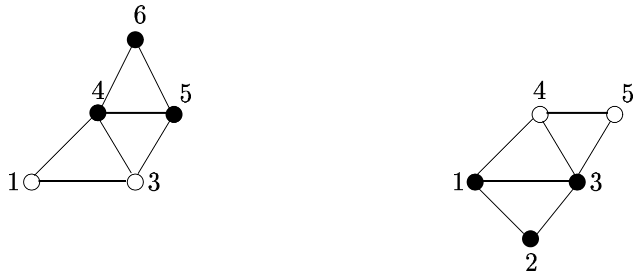
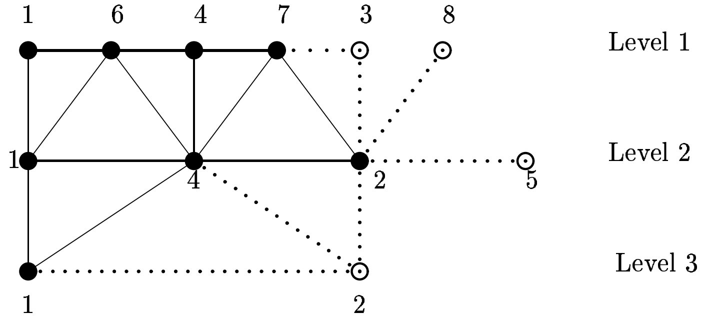


Figure 3 Example of a 1D grid with three levels of refinement. The ghost-nodes complete the intra-grid and inter-grid connections. The full-nodes are drawn as dark circles while the ghost-nodes are drawn as open circles.



inter-grid connections. See Figure 3. This is the basis of our parallel implementation of the multigrid method.

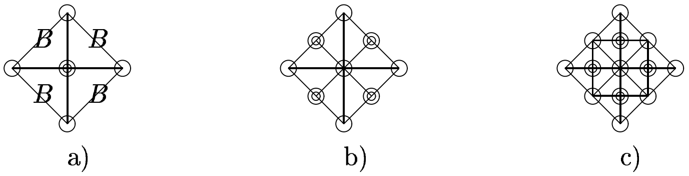
In order to communicate any updates, the full-nodes must know which processors contain any corresponding ghost-node and each ghost-node must know which processor contains the corresponding full-node. This information is stored in the neighbour node table.

The use of ghost-nodes as a communication buffer or as a way of storing updates from neighbouring processors is not new, see for example the ghost cells used in [BKFF94] and artificial boundary in [MFL$^+$91]. However, we have extended their application so that they also define the data dependencies. For example, during refinement the communication pattern has to be updated when new nodes are added and by exploiting the relationship between the ghost-nodes and full-nodes this can be done independently across the processors.

4 Refinement

Our method of refinement is based upon the newest node bisection method. This method refines the triangles by splitting the edges which sit opposite the newest nodes. See [Mit88], [Mit89], [Mit92] and [Sta95] for a more detailed discussion. In the node-edge data structure it is easier to work with the base edges rather then the newest nodes. The base edges are the edges which sit opposite the newest nodes. Figure 4 shows an example.

Figure 4 Example refinement using newest node bisection. Figure a) shows the initial grid. We assume that the centre node is our initial 'newest node'. The corresponding bases edges are marked by a B. Figure b) shows the result after one refinement sweep. Figure c) shows the final grid.



When studying the mechanics behind the refinement we see that the most difficult part is adding the nodes. When a new node is added the algorithm must determine which processor should get the new node as a full-node and which, if any, should get it as a ghost-node. By exploiting the dependencies within the data structure a set of rules can be developed which lets each processor resolve this problems without the need for any communication.

In his thesis ([Mit88]), Mitchell describes a method of adaptive refinement which uses bisection and compatibly divisible triangles. The disadvantage of this method, from our point of view, is that it only works on one triangle at a time. We have extended Mitchell's method so that several triangles may be refined at once by using interface-base edges. Interface-base edges are edges which sit between two different levels of refinement. For example, in Figure 5 a) we have marked the interface-base edges by an I and the base edges by a B. The neighbouring coarse triangles must be refined before the interface-base edges are split. So if the edge marked by I_1 needed to be split, then the base edge B_3 must be split first as shown in Figure 5 b). Note that the interface-base edge I_1 has been updated to a base edge B_1. The edge B_1 can now be split to give the final grid shown in Figure 5 c).

By using this approach it is possible to split more then one triangle at a time. In Figure 5 a) I_2 could be split the same time as I_1.

Keeping track of the interface-base edges guarantees that the angles remain bound away from 0 and π. The disadvantage is that several of the neighbouring coarse grid triangles may need to be refined, in Figure 5 c) edges B_4 and I_8 must be split before I_{11}, and hence the refinement may travel across several processors. The algorithm uses communication to control the order of refinement around the boundary of the grid in each processor.

5 Load Balancing

After a refinement sweep (whole grid or adaptive) we may find that the load needs to be rebalanced. Rebalancing the load is a deceptively difficult problem. We only give a very brief overview of our four step heuristic algorithm, but this hides a lot of the detail and the subtle problems which arise in the actual implementation.

To re-balance the load we let the nodes 'flow' out of the processors with too many nodes into the processors which do not have enough. By flow we mean that the nodes follow the edges between neighbouring processors.

The algorithm consists four steps. The first step calculates how many nodes should be moved in order to balance the load, the second steps picks which nodes should be moved, the third step finds which processors the nodes should be moved to and the final step moves the nodes.

More information is given in [Sta95].

6 Example Runs

The program is written in a mixture of C^{++} and PVM.

The results given in this paper were obtained on the Fujitsu AP1000. The AP1000 is a distributed memory MIMD machine with 128 processors arranged in a 2D torus. Each processor uses a 25MHz SPARC chip. For further information see [Aus91], [Aus92], [Fuj90], [Haw91] and [IHI$^+$].

The example we shall consider solves the equation $-\Delta u+u=-e^x e^y$ on the octagon shown in Figure 6, with the boundary condition chosen so that the exact solution is $u=e^x e^y$. The results are given in table 1.

The whole grid was refined to the given number of levels by using the newest node bisection method. Two iterations of the V-scheme were applied with two pre and two post smoothers. The efficiency results are calculated as $T_1/(p\,T_p) \times n_p/n_1$, where T_p is the total time for p processors and n_p is the number of nodes for p processors.

The time has been broken up into the four major modules; the FEM module which calculate the stiffness matrix, the V-scheme module which solves the system of equations, the Refine module which refines the grid and the Load module which balances the load.

The efficiency for the FEM module is very high. The ghost-nodes have been used to complete the connections, so this module does not need to do any communication.

The efficiency for the Refine module is also high. By exploiting the relationship between the ghost-nodes and full-nodes we were able to refine the grids in parallel without any communication.

The efficiency for the V-scheme module drops off for large number of processor. This is as expected since the V-scheme module spends more time in the coarse grids then the other modules. Note that the coarsest grid only contains nine nodes so there will be a lot of idle processors when doing computations on the coarse grids.

The overall efficiency decreases as we increase the number of processors. However, we can see from the times for the Load module that most of the increased cost comes from spreading the grids across the processors (after each new level of refinement is built we spread the grid out to fill up as many processors as possible). Once we have

Table 1 Efficiency results for $-\Delta u + u = -e^x e^y$ on a octagon domain.

No. of Processors	1	4	16	64
No. of Levels	5	6	7	8
No. of Nodes	4225	16641	66049	263169
Total (sec)	61.0	64.3	81.7	129.0
V-scheme (sec)	13.0	15.2	20.1	42.3
FEM (sec)	31.5	32.0	34.6	30.2
Refine (sec)	16.1	14.6	14.6	11.0
Load (sec)	0.0	3.3	19.0	67.9
Efficiency (%)		93	73	46

Table 2 The efficiency results for $-\Delta u = \sin(\pi x)\sin(\pi y)$ on the unit square domain. Note that the coarse grid size is increased as the number of processors is increased.

No. of Processors	1	4	16	64
No. of Levels	6	7	8	9
No. of Fine Nodes	4225	16641	66049	263169
No. of Coarse Nodes	81	289	1089	4225
Total (sec)	51.0	50.9	54.5	56.7
V-scheme (sec)	12.9	14.3	16.4	17.3
FEM (sec)	22.1	22.7	23.9	24.4
Refine (sec)	15.6	12.7	11.2	12.3
Load (sec)	0.0	1.3	5.5	6.3
Efficiency (%)		99	91	88

enough nodes to fill the processors the efficiency increases markedly. To verify this statement, we tried another example where the coarse grid size was also increased as the number of processors was increased. Table 2 gives the results for solving the equation $-\Delta u = \sin(\pi x)\sin(\pi y)$ on the unit square domain.

We have recently started working on non-linear problems. Figures 7 and 8 shows the result after solving the equation $-\Delta u - 2e^{-5\times10^4} = -2$ using four levels of adaptive refinement. The domain is as shown in Figure 7 with $u = 1$ on the inner boundary and $u = 0$ on the outer boundary. As this problem is more computationally expensive then the previous example, the initial cost of setting up the grid (i.e. the cost of the Load Routine) is less significant. On a network of workstations the Load Routine took less then 5% of the overall time.

REFERENCES

[Aus91] Australian National University, Department Of Computer Science, Canberra, ACT, 0200, Australia (November 1991) *Proceedings Of The Second Fujitsu-ANU CAP Workshop.*

[Aus92] Australian National University, Department Of Computer Science, Canberra, ACT, 0200, Australia (March 30 1992) *AP1000 User's Guide.*

[BKFF94] Baden S. B., Kohn S. R., Figueira S. M., and Fink S. J. (11 April 1994) The LPARX user's guide, v 1.0. Technical report, Department of Computer Science and Engineering, University of California, San Diego, La Jolla, CA 92093-0114 USA.

[Fuj90] Fujitsu Laboratories Ltd., Computer Based Systems Lab. Kawasaki. Fujitsu Laboratories Ltd. 1015 Kamikondanaka, Nakahara-ku, Kawasaki 211, Japan (March 1990) *Cap-II Program Development Guide [1]: C-Language Interface*, second edition.

[Haw91] Hawking D. (May 1991) About the Fujitsu AP1000. Technical report, Department Of Computer Science, Australian National University, Canberra, ACT 0200, Australia.

[IHI$^+$] Ishihata H., Horie T., Inano S., Shimizu T., and Kato S.*CAP-II Architecture.* Computer Based Systems Lab. Kawasaki. Fujitsu Laboratories Ltd. 1015 Kamikodanaka, Nakahara-ku, Kawasaki 211, Japan.

[MFL$^+$91] McBryan O. A., Frederickson P. O., Linden J., Schüller A., Solchenbach K., Stüden K., Thole C., and Trottenberg U. (1991) Multigrid methods on parallel computers - A survey of recent developments. *IMPACT Comput. Sci. Engng.* 3: 1–75.

[Mit88] Mitchell W. F. (1988) *Unified Multilevel Adaptive Finite Element Methods For Elliptic Problems.* PhD thesis, Department Of Computer Science, University Of Illinois at Urbana-Champaign, Urbana, IL. Technical Report UIUCDCS-R-88-1436.

[Mit89] Mitchell W. F. (December 1989) A comparison of adaptive refinement techniques for elliptic problems. *ACM Trans. Math. Software* 15(4): 326–347.

[Mit92] Mitchell W. F. (January 1992) Optimal multilevel iterative methods for adaptive grids. *SIAM J. Sci. Stat. Comput* 13(1): 146–167.

[Rüd92] Rüde U. (May 1992) Data structures for multilevel adaptive methods and iterative solvers. Technical Report I-9217, Institut für Informatik, TU München. Copy found in ftp: capser.cs.yale.edu, dir: mgnet/papers/Ruede, file: data_struct.∗.

[Rüd93a] Rüde U. (1993) Data abstraction techniques for multilevel algorithms. In *Proceedings of the GAMM-Seminar on Multigrid Methods.* Institut für Angewandte Analysis und Stochastik. Copy found in ftp: capser.cs.yale.edu, dir: mgnet/papers/Ruede, file: programming.∗.

[Rüd93b] Rüde U. (1993) *Mathematical and computational techniques for multilevel adaptive methods.* SIAM.

[Sta95] Stals L. (1995) *Parallel Multigrid On Unstructured Grids Using Adaptive Finite Element Methods.* PhD thesis, Department Of Mathematics, Australian National University, Canberra, 0200, Australia.

Figure 5 Example of adaptive refinement. In Figure a) I_1 and I_2 are two interface edges while B_3 and B_4 are two base edges. Note that we have not marked all of the base and interface-base edges to help to reduce the clutter. The base edge B_3 must be split before the interface-base edge I_1. When B_3 is split the interface-base edge I_1 is updated to a base edge B_1 as shown in b). The edge B_1 can now be split to give the final grid shown in c).

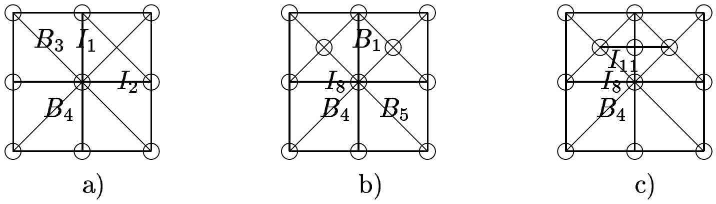


Figure 6 Example grid spread over four processor after three levels of refinement. The areas which are not shaded are shared by two or more processors.

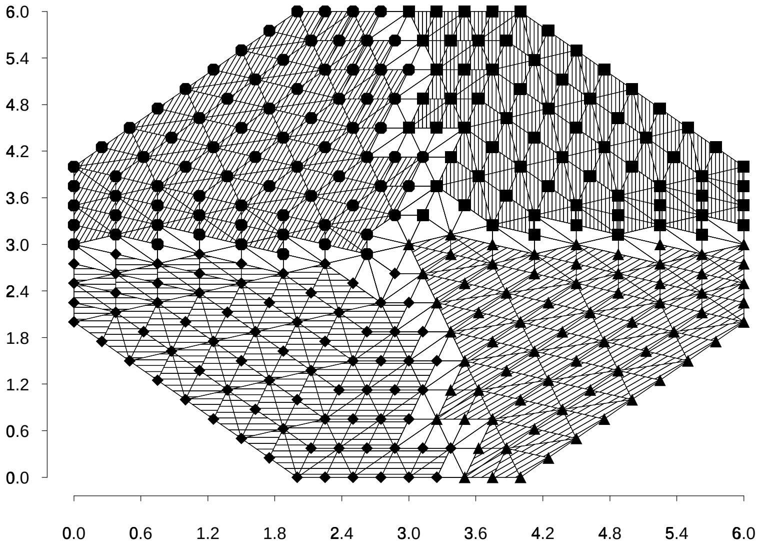


Figure 7 Resulting grid after four levels of adaptive refinement. This example was run a network of workstations using eight processors.

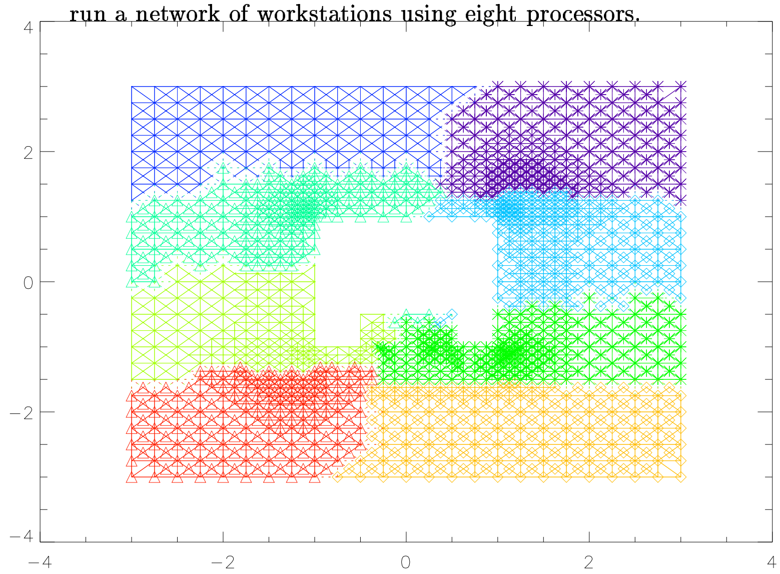
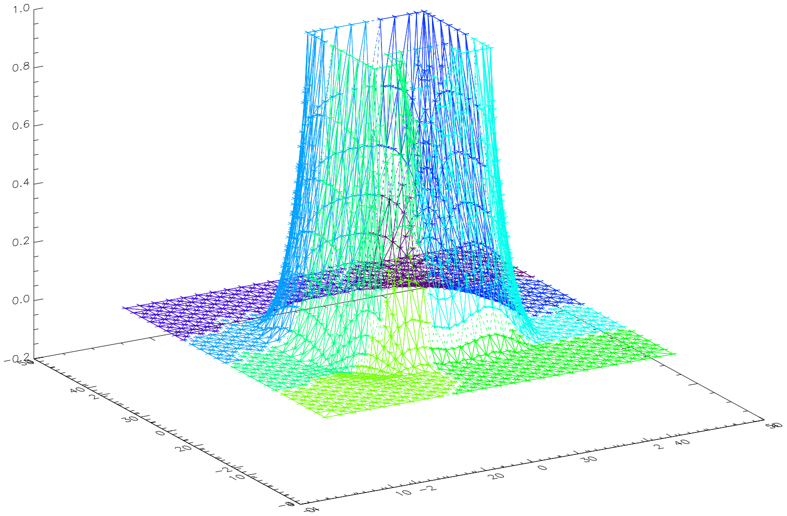


Figure 8 Result after four levels of adaptive refinement.



59

Hierarchical Boundary Element Preconditioners in Domain Decomposition Methods

O. Steinbach and W. L. Wendland

1 Introduction

For a non–overlapping domain decomposition of a bounded domain $\Omega \subset I\!R^n$ $(n = 2, 3)$ we consider a variational problem to find $u \in V$ such that

$$a(u,v) \;=\; f(v) \tag{1.1}$$

holds for all test functions $v \in V$. This formulation corresponds to a mixed boundary value problem for a self–adjoint and elliptic partial differential operator. The Hilbert space V is given by all functions $u \in H^{1/2}(\Gamma_S)$ vanishing on the Dirichlet boundary Γ_D; Γ_S denotes the skeleton of the domain decomposition and $f(\cdot)$ is a given bounded linear form. The symmetric and V–elliptic bilinear form in (1.1) is given by

$$a(u,v) \;=\; \sum_{i=1}^{p} \int\limits_{\Gamma_i} (S_i u_{|\Gamma_i})(x) \cdot v_{|\Gamma_i}(x)\, ds_x \;, \tag{1.2}$$

where S_i denotes the locally defined Steklov–Poincaré operators mapping the local Dirichlet data $u_{|\Gamma_i}$ onto the Neumann data t_i. This Dirichlet–Neumann map can be expressed explicitly by boundary integral operators in terms of the boundary integral equations

$$\left.\begin{array}{rcl} (V_i t_i)(x) & = & (\frac{1}{2}I + K_i)u_{|\Gamma_i}(x) - (N_0^i f)(x), \\ (D_i u_{|\Gamma_i})(x) & = & (\frac{1}{2}I - K_i')t_i(x) - (N_1^i f)(x) \end{array}\right\} \text{for } x \in \Gamma_i. \tag{1.3}$$

The mapping properties of all operators introduced above are well known [Cos88]. The symmetric representation of S_i follows immediately from (1.3),

$$(S_i u_{|\Gamma_i})(x) \;=\; \left[D_i + \tilde{K}_i' V_i^{-1} \tilde{K}_i\right]\; u_{|\Gamma_i}(x) + N_1^i f - \tilde{K}_i' V_i^{-1} N_0^i f\,, \tag{1.4}$$

where

$$(\tilde{K}_i u_{|\Gamma_i})(x) \;=\; \begin{cases} (K_i u_{|\Gamma_i})(x) & \text{for } x \in \Gamma_D, \\ (\frac{1}{2}I + K_i)u_{|\Gamma_i}(x) & \text{elsewhere.} \end{cases}$$

Let $V_h \subset V$ be a finite dimensional subspace, then the Galerkin–Bubnov discretization of (1.1) based on the symmetric formulation (1.4) leads to the algebraic system of linear equations

$$\sum_{i=1}^{p} A_i^\top \left(D_{h,i} + \tilde{K}_{h,i}^\top V_{h,i}^{-1} \tilde{K}_{h,i} \right) A_i \underline{u} \;=\; \underline{f}\,. \tag{1.5}$$

The local stiffness matrices are given by

$$\begin{aligned} D_{h,i}[\ell, k] \;&=\; \langle D_i {}'^{\mu}_{k}, {}'^{\mu}_{\ell} \rangle_{L^2(\Gamma_i)}\,, \\ \tilde{K}_{h,i}[s, k] \;&=\; \langle \tilde{K}_i {}'^{\mu}_{k}, {}'^{\nu}_{s} \rangle_{L^2(\Gamma_i}\,, \\ V_{h,i}[s, r] \;&=\; \langle V_i {}'^{\nu}_{r}, {}'^{\nu}_{s} \rangle_{L^2(\Gamma_i)} \end{aligned}$$

for $k, \ell = 1, \dots, N_i$, $r, s = 1, \dots, M_i$ and where A_i denotes Boolean matrices describing the transformation of the global numbering into the local one. The ${}'^{\mu}_{k}$ and ${}'^{\nu}_{r}$ are appropriate trial functions, e.g. smoothest piecewise polynomial B–splines of degree μ and ν, respectively. To solve the symmetric and positive definite system (1.5) by the conjugate gradient iteration scheme we need an optimal preconditioner to keep the numerical amount of work as low as possible. The construction of a preconditioner is essential for domain decomposition algorithms based either on a finite element or a boundary element discretization of the original problem as well as on coupling both. There are numerous different approaches to solve the corresponding finite element equations by using hierarchical [BPS87, SBG96, Wid88] or of Neumann–Neumann type preconditioners [LeT94] or [HW92] with boundary elements. However, the resulting spectral condition number of the preconditioned system matrix often depends on mesh and material parameters of the model. Here we give a general technique to construct optimal preconditioners independent of these bad parameters by using the symmetric representation of the local Steklov–Poincaré operators and its spectral equivalence to the Galerkin discretization of the hypersingular integral operator.

2 Spectral Equivalence Inequalities

To construct an optimal preconditioner C_S for the assembled stiffness matrix

$$S_h \;=\; \sum_{i=1}^{p} A_i^\top S_{h,i} A_i \tag{2.6}$$

we first consider the local matrices

$$S_{h,i} \;=\; D_{h,i} + \tilde{K}_{h,i}^\top V_{h,i}^{-1} \tilde{K}_{h,i}\,. \tag{2.7}$$

Obviously, we have the lower spectral equivalence inequality

$$(D_{h,i}\underline{v}_i, \underline{v}_i) \;\leq\; (S_{h,i}\underline{v}_i, \underline{v}_i) \tag{2.8}$$

for all $\underline{v}_i \in I\!R^{N_i}$ due to the $H^{-1/2}(\Gamma_i)$–ellipticity of the single layer potential operators. For two–dimensional problems suppose $\operatorname{diam}(\Omega_i) < 1$ [HW77]. Since V_i is a self–adjoint pseudodifferential operator, the operator $T_i = \tilde{K}_i' V_i^{-1} \tilde{K}_i$ is self–adjoint and $H^{1/2}(\Gamma_i)$–elliptic, too. If we denote by $T_{h,i}$ the Galerkin discretization with the matrix entries

$$T_{h,i}[\ell, k] = \langle T_i \psi_k^\mu, \psi_\ell^\mu \rangle_{L^2(\Gamma_i)}$$

for $k, \ell = 1, \ldots, N_i$, we get the upper spectral equivalence inequality [Ste96]

$$(\tilde{K}_{h,i}^\top V_{h,i}^{-1} \tilde{K}_{h,i} \underline{v}_i, \underline{v}_i) \le (T_{h,i} \underline{v}_i, \underline{v}_i) \tag{2.9}$$

due to the ellipticity of V_i, and by adding $(D_{h,i}\underline{v}_i, \underline{v}_i)$,

$$(S_{h,i}\underline{v}_i, \underline{v}_i) \le ((D_{h,i} + T_{h,i})\underline{v}_i, \underline{v}_i) \tag{2.10}$$

for all $\underline{v}_i \in I\!R^{N_i}$. This means, that the discrete Steklov–Poincaré operator $S_{h,i}$ is bounded by the Galerkin discretization of the continuous Steklov–Poincaré operator. Since S_i and D_i are both $H^{1/2}(\Gamma_S)$ semi–definite and bounded, the discrete Steklov–Poincaré operator $S_{h,i}$ is spectrally equivalent either to $D_{h,i} + T_{h,i}$ or to $D_{h,i}$. This result holds independent of the dimension and the discretization, i.e. of the mesh and trial functions used. Altogether we have the spectral equivalence inequalities

$$\sum_{i=1}^{p} (D_{h,i}\underline{v}_i, \underline{v}_i) \le \sum_{i=1}^{p} (S_{h,i}\underline{v}_i, \underline{v}_i) \le \sum_{i=1}^{p} ((D_{h,i} + T_{h,i})\underline{v}_i, \underline{v}_i) \tag{2.11}$$

with $\underline{v}_i = A_i \underline{v}$. Employing the isomorphism $\underline{v} \in I\!R^N \leftrightarrow v_h \in H^{1/2}(\Gamma_S)$, this is also equivalent to

$$c_1 \cdot \sum_{i=1}^{p} ||v_h||^2_{H^{1/2}(\Gamma_i)} \le \sum_{i=1}^{p} (S_{h,i}\underline{v}_i, \underline{v}_i) \le c_2 \cdot \sum_{i=1}^{p} ||v_h||^2_{H^{1/2}(\Gamma_i)}$$

where the constants are independent of the discretization parameters.

Let us denote by v_I^i the piecewise linear interpolant of a function $v^i \in H^{1/2}(\Gamma_i)$ with $v_I^i(x_C) = v^i(x_C)$ for all coarse grid nodes x_C associated with a mesh size H_i. Then from Sobolev's imbedding theorem, one obtains the error estimate

$$||v^i - v_I^i||^2_{H^{1/2}(\Gamma_i)} \le c \cdot L(s) \cdot H_i^{2s-1} \cdot ||v^i||^2_{H^s(\Gamma_i)} \tag{2.12}$$

for $s > \frac{n-1}{2}$, where $L(s) = (2s - n + 1)^{1-n}$. For a function $v_h \in V_h$ and $v_h^i = v_{h|\Gamma_i}$, this inequality, together with the inverse inequality in V_h, implies the stability condition

$$||v_h^i - v_I^i||^2_{H^{1/2}(\Gamma_i)} \le c \cdot K(h_i, H_i) \cdot ||v_h^i||^2_{H^{1/2}(\Gamma_i)} \tag{2.13}$$

where

$$K(h_i, H_i) = \gamma^{n-2} \cdot (\log \gamma)^{n-1}, \ \gamma = \frac{H_i}{h_i} \, .$$

For the bilinear form

$$c(v, v) = \sum_{i=1}^{p} \left\{ ||v - v_I||^2_{H^{1/2}(\Gamma_i)} + ||v_I||^2_{H^{1/2}(\Gamma_i)} \right\} \tag{2.14}$$

then we find the spectral equivalence inequalities

$$\frac{c_1}{(1+K(h,H))} \cdot c(v_h,v_h) \;\le\; \sum_{i=1}^{p} (S_{h,i}\underline{v}_i, \underline{v}_i) \;\le\; c_2 \cdot c(v_h,v_h)\;, \tag{2.15}$$

which correspond to the spectral equivalence inequalities for finite element preconditioners of hierarchical type, c.f. [BPS87, Wid88]. According to (2.14) and the mapping properties of the hypersingular integral operators, the preconditioning bilinear form is given by

$$\tilde{c}(u,v) \;=\; \sum_{i=1}^{p} \left\{ \langle D_i(u-u_I), v-v_I \rangle_{L^2(\Gamma_i)} \;+\; \langle D_i u_I, v_I \rangle_{L^2(\Gamma_i)} \right\}\;. \tag{2.16}$$

3 Preconditioners

For a given function $v_h \in V_h \leftrightarrow \underline{v} \in I\!R^N$ the splitting

$$v_h(x) \;=\; \tilde{v}_h + v_I(x), \quad \tilde{v}_h(x) = v_h(x) - v_I(x)$$

corresponds to the basis transformation

$$\begin{pmatrix} \underline{v}_H \\ \underline{\tilde{v}} \end{pmatrix} = \begin{pmatrix} I_q & 0 \\ -I_h & I_{N-q} \end{pmatrix} \underline{v}\,. \tag{3.17}$$

Here, I_q is the identity matrix of dimension q corresponding to the number of unknown coarse grid nodes; I_{N-q} is the identity matrix for all remaining fine grid nodes and I_h is the discrete counterpart of the linear interpolation. The Galerkin discretization of the preconditioning form (2.16) now leads to the matrix representation

$$C_{S,1} \;=\; \begin{pmatrix} I_q & -I_h^\top \\ 0 & I_{N-q} \end{pmatrix} \begin{pmatrix} D_{HH} & 0 \\ 0 & D_{hh} \end{pmatrix} \begin{pmatrix} I_q & 0 \\ -I_h & I_{N-q} \end{pmatrix}, \tag{3.18}$$

where D_{HH} and D_{hh} are the assembled stiffness matrices for the coarse and fine grid trial functions, respectively. In general, this preconditioner corresponds to the BPS preconditioner [BPS87], which was also used for elasticity problems in [SBG96]. However, the diagonal matrix in (3.18) may be computed exactly by using the hypersingular boundary integral operator. If we have given a fine grid preconditioner C_h, which is spectrally equivalent to D_{hh}, then the resulting hierarchical preconditioner is given by

$$C_{hier}^{-1} \;=\; \begin{pmatrix} I_q & 0 \\ I_h & I_{N-q} \end{pmatrix} \begin{pmatrix} D_{HH}^{-1} & 0 \\ 0 & C_h^{-1} \end{pmatrix} \begin{pmatrix} I_q & I_h^\top \\ 0 & I_{N-q} \end{pmatrix}. \tag{3.19}$$

Due to the spectral equivalence inequalities (2.15), the spectral condition number of the preconditioned system is bounded by

$$\kappa(C_{hier}^{-1} S_h) \;\le\; c \cdot (1 + K(h,H))\;,$$

which depends on the discretization parameters, i.e. on the relation of the coarse to the fine grid mesh sizes.

Since this preconditioner (3.19) is not optimal, we consider a second one. For the matrix

$$S_{h,2} = \sum_{i=1}^{p} A_i^\top D_{h,i} A_i \tag{3.20}$$

we conclude from (2.11), that the spectral condition number of the preconditioned system $C_{S,2}^{-1} S_h$ is bounded by a constant, i.e.

$$\kappa(C_{S,2}^{-1} S_h) \le c,$$

where c does not depend on the discretization parameters and not on the domain decomposition considered, either. On the other hand, we have to realize the matrix multiplication with $C_{S,2}^{-1}$ in an efficient manner. Since the matrix $C_{S,2}$ is given explicitly by the locally stored matrices $D_{h,i}$, the matrix times vector multiplications can be executed in parallel. Therefore we can use iterative schemes to realize $C_{S,2}^{-1}$, e.g. multigrid methods [CKL96] or a conjugate gradient iteration using the BPS type preconditioner described above.

4 Numerical Results

In this section we compare our proposed preconditioner $C_{S,2} = CG(D_h)$ with the known BPS preconditioner in the case of the Laplace equation. We further show, that this technique can be used also in the case of linear elasticity. We compare the number of cg iterations and the corresponding computing times to get a relative error reduction of 10^{-6}. All computations were made on an Intel Paragon.

For the simple model problem of the Laplace equation in the unit square and a domain decomposition into 64 subdomains we get the results shown in Table 1.1:

Table 1 Numerical results for the Laplace equation

	BPS		$CG(D_h)$	
N	Iter	sec	Iter	sec
64	24	22.52	11	41.28
128	24	26.55	12	47.01
256	25	45.47	13	62.11
512	27	127.39	13	107.80
1024	28	489.38	14	326.77

The number of iterations for the BPS preconditioner is twice the number of our proposed technique, where, on the other hand, the costs to realize the preconditioner are more expensive. Since we can bound the spectral condition number independent of the mesh size, and since the costs of the preconditioners are to set in relation to the matrix times vector multiplication itself, i.e. the solution of a mixed boundary value problem per global iteration step for the realization of the Steklov–Poincaré operator, our new preconditioner seems to be optimal, in agreement with our theory.

In Table 1.2 we present the results of our proposed preconditioner by solving a mixed boundary value problem in linear elasticity with up to 32 subdomains. As one can see, the number of iterations is nearly the same as for the Laplace equation, i.e. we have independence of the underlying partial differential equation.

Table 2 Numerical results in linear elasticity

	p=2		p=8		p=32	
N	Iter	sec	Iter	sec	Iter	sec
64	11	3.49	13	11.78	14	111.82
128	11	9.87	14	26.08	15	139.82
256	11	33.56	14	73.43	15	205.59
512	11	129.22	14	263.63	15	438.44

5 Conclusions

The proposed preconditioning technique is based on the Galerkin discretization of the hypersingular boundary integral operator; and therefore is well suited for the symmetric formulation of boundary element methods. We note, that the Galerkin discretization of the hypersingular integral operator can be reduced to the computation of weakly singular integral operators by partial integration [Ned82]. Because of the spectral equivalence of the discrete Steklov–Poincaré operator and the discrete hypersingular operator, from the latter one can derive other preconditioners of algebraic type, as e.g. multigrid methods or structured matrices like block circulant matrices, which, in turn, can be inverted by the fast Fourier transformation. The realization of the matrix times vector multiplication with the discrete Steklov–Poincaré operator requires the solution of local mixed boundary value problems. For the iterative solution of these problems one can use the concept of pseudodifferential operators of dual order [SW96], i.e. the discrete hypersingular integral operator can be used as a preconditioner of the single layer potential and vice versa. The proposed preconditioning technique is almost independent of the underlying partial differential equation and is also well suited for coupled boundary and finite element methods [Lan94].

REFERENCES

[BPS87] Bramble J. H., Pasciak J. E., and Schatz A. H. (1987) The construction of preconditioners for elliptic problems by substructuring I. *Math. Comp.* 47(175): 103–134.

[CKL96] Carstensen C., Kuhn M., and Langer U. (1996) Fast parallel solvers for symmetric boundary element domain decomposition methods. Institutsbericht 500, Universität Linz.

[Cos88] Costabel M. (1988) Boundary integral operators on Lipschitz domains: Elementary results. *SIAM J. Math. Anal.* 19(3): 613–626.

[HW77] Hsiao G. C. and Wendland W. L. (1977) A finite element method for some integral equations of the first kind. *J. Math. Anal. Appl.* 58: 449–481.

[HW92] Hsiao G. C. and Wendland W. L. (1992) Domain decomposition via boundary element methods. In Alder H. (ed) *Numerical Methods in Engineering and Applied Sciences*, pages 198–207. CIMNE, Barcelona.

[Lan94] Langer U. (1994) Parallel iterative solution of symmetric coupled fe/be–equations via domain decomposition. *Contemp. Math.* 157: 335–344.

[LeT94] LeTallec P. (1994) Domain decomposition methods in computational mechanics. *Comput. Mech. Adv.* 1(2): 121–220.

[Ned82] Nedelec J. C. (1982) Integral equations with non integrable kernels. *Integral Equations Oper. Theory* 5: 562–572.

[Smi91] Smith B. F. (1991) *Domain Decomposition Algorithms for the Partial Differential Equations of Linear Elasticity.* PhD thesis, New York University.

[Ste96] Steinbach O. (1996) *Gebietszerlegungsmethoden mit Randintegralgleichungen und effiziente numerische Lösungsverfahren für gemischte Randwertprobleme.* PhD thesis, Universität Stuttgart.

[SW96] Steinbach O. and Wendland W. L. (1996) Efficient preconditioners for boundary element methods and their use in domain decomposition methods. In Glowinski R., Périaux J. P., Shi Z. C., and Widlund O. B. (eds) *Proc. Eightth Int. Conf. on Domain Decomposition Meths.* Wiley and Sons, Chichester.

[Wid88] Widlund O. (1988) Iterative substructuring methods: Algorithms and theory for elliptic problems in the plane. In Glowinski R., Golub G. H., Meurant G. A., and Périaux J. (eds) *Domain Decomposition Methods for Partial Differential equations. Proceedings of the First International Conference on Domain Decomposition Methods*, pages 113–128. Philadelphia.

60

Two-level Schwarz Methods for Indefinite Integral Equations

M. Maischak, Ernst P. Stephan and Thanh Tran

1 Introduction

In this paper we consider additive Schwarz preconditioners for indefinite linear systems arising from the h-version of the boundary element method (BEM) for solving Helmholtz problems. Here we extend the approach introduced by Cai and Widlund [CW92] for finite element discretizations to boundary element discretizations. We report on two-level methods applied to the h-version of the Galerkin method for weakly singular and hypersingular integral equations of the first kind on the interval $\Gamma = (-1, 1)$. The Neumann problem for the Helmholtz equation in $\mathbb{R}^2 \backslash \bar{\Gamma}$ leads to the hypersingular integral equation

$$D_k v(x) := -\frac{i}{2} \frac{\partial}{\partial n_x} \int_\Gamma \frac{\partial}{\partial n_y} [H_0^1(k|x-y|)] v(y)\, ds_y = g_1(x), \quad x \in \Gamma. \tag{1.1}$$

Correspondingly the Dirichlet problem leads to the weakly singular integral equation

$$V_k \psi(x) = \int_\Gamma H_0^1(k|x-y|) \psi(y)\, ds_y = g_2(x), \quad x \in \Gamma. \tag{1.2}$$

There H_0^1 is the Hankel function of the first kind and of order zero, $\mathrm{Im}k \geq 0$, $k \neq 0$ and $\frac{\partial}{\partial n}$ denotes the normal derivative on Γ.

It was shown in [SW90, SW84] that for $g_1 \in H^{-1/2}(\Gamma)$ equation (1.1) has a unique solution in $\tilde{H}^{1/2}(\Gamma) := H_{00}^{1/2}(\Gamma)$ whereas for given $g_2 \in H^{1/2}(\Gamma)$ equation (1.2) has a unique solution in $\tilde{H}^{-1/2}(\Gamma)$ (the dual of $H^{1/2}(\Gamma)$). (For definitions of the Sobolev spaces see [LM72]). Note that D_k is a pseudodifferential operator B_α of order $\alpha = 1$ and V_k of order $\alpha = -1$ mapping $\tilde{H}^\alpha(\Gamma)$ into $H^{-\alpha/2}(\Gamma)$ both satisfying $B_\alpha = A_\alpha + K_\alpha$ with a positive definite operator A_α and a compact operator K_α.

With $f = g_1$ in (1.1) and $f = g_2$ in (1.2) the boundary element Galerkin schemes for the above integral equations ($\alpha = 1$ or $\alpha = -1$) read as follows:
For a given subspace X_N^α of $\tilde{H}^{\alpha/2}$ find $u_N \in X_N^\alpha$ such that

$$\langle B_\alpha u_N, \phi \rangle_{L^2(\Gamma)} = \langle f, \phi \rangle_{L^2(\Gamma)} \quad \text{for all } \phi \in X_N^\alpha. \tag{1.3}$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes ©1998 DDM.org

These Galerkin schemes lead to very large indefinite systems of linear equations with dense and ill-conditioned system matrices and therefore iterative methods require good preconditioners [ST97a, ST97b, MS97, MST97]. We report on additive Schwarz methods applied to (1.3) which are efficient preconditioners for the GMRES method. For efficient Schwarz preconditioners for positive definite boundary integral equations see [TS97, HS96, Ste96].

2 Preconditioners for the Hypersingular Operator

As subspace X^1_N we use the subspace $S^1_h(\Gamma)$ of continuous, piecewise linear functions on a quasi-uniform mesh which vanish at the endpoints of Γ. Let ϕ^h_j, $j = 1, \dots, N-1$ denote the hat function which takes value 1 at the meshpoint x_j and 0 at other mesh points. These functions form a basis for $S^1_h(\Gamma)$. We then decompose $S = S^1_h(\Gamma)$ as

$$S = S_0 + S_1 + \dots + S_{N-1} \tag{2.4}$$

where $S_0 = S^1_H(\Gamma)$ is defined as $S^1_h(\Gamma)$ with mesh size $H = 2h$ and $S_j = \text{span}\,\{\phi^h_j\}$ for $j = 1, \dots, N-1$.

Let operators Q_j be defined via

$$\langle B_1 Q_j w, v_j \rangle = \langle B_1 w, v_j \rangle \quad \forall w \in S, v_j \in S_j, j = 0, 1, \dots, N-1. \tag{2.5}$$

Then the additive Schwarz operator is given by $Q = Q_0 + \dots + Q_{N-1}$ and the additive Schwarz method consists in solving

$$Qu_N = b_N \tag{2.6}$$

with RHS $b_N = \sum_{j=0}^{N} b_j$ where

$$\langle b_j, v_j \rangle = \langle f, v_j \rangle \; \forall v_j \in S_j, \; j = 0, 1, \dots, N-1. \tag{2.7}$$

Then as shown in [ST97a] this algorithm when used with the GMRES method gives an efficient solver for the Galerkin scheme (1.3), namely the rates of convergence of the Schwarz operator is bounded from above independently of the number of degrees of freedom if the mesh size of the coarse space S_0 is sufficiently small. As proved in [CW92] the rate of convergence of the GMRES method when used to solve (2.6) is given as $1 - \frac{C_0^2}{C_1^2}$ where

$$C_0^2 = \inf_{v \in S} \frac{\langle A_1 v, Qv \rangle}{\langle A_1 v, v \rangle} \text{ and } C_1^2 = \sup_{v \in S} \frac{\langle A_1 Qv, Qv \rangle}{\langle A_1 v, v \rangle}. \tag{2.8}$$

In view of this result we show in [ST97a] that C_0 and C_1 are independent of the number of degrees of freedom.

To get rid of a large coarse subspace S_0 we consider in [MS97] a non-overlapping method where one has a coarse mesh which is almost independent of the fine mesh.

The coarse mesh: We divide Γ into disjoint subdomains Γ_i, $i = 1, \dots, J$, so that $\bar{\Gamma} = \cup_{i=1}^{J} \bar{\Gamma}_i$. The length of Γ_i is denoted by H_i.

The fine mesh: We further divide each Γ_i into disjoint subintervals Γ_{ij}, $j = 1, \ldots, N_i$, so that $\bar{\Gamma}_i = \cup_{j=1}^{N_i} \bar{\Gamma}_{ij}$. The maximum length of the subintervals in Γ_i is denoted by h_i. For the non-overlapping method, we require that the fine mesh is locally quasi-uniform, i.e., it is quasi-uniform in each subdomain.

The additive Schwarz method is designed via an appropriate decomposition of $S_h(\Gamma)$

$$S_h(\Gamma) = S_H(\Gamma) \oplus \bigoplus_{i=1}^{J} \bigoplus_{j=1}^{N_i} S_1^0(\Gamma_{ij}) \tag{2.9}$$

where

$$S_H(\Gamma) := \{v \in C(\Gamma) : v|_{\Gamma_i} \in \mathcal{P}_1(\Gamma_i) \text{ for } i = 1, \ldots, J;\ v(\pm 1) = 0\}$$
$$S_1^0(\Gamma_{ij}) := \{v \in \mathcal{P}_1(\Gamma_{ij}) : v = 0 \text{ at the endpoints of } \Gamma_{ij}\},\ i = 1, \ldots, J,\ j = 1, \ldots, N_i.$$

Then the corresponding algorithm consists in solving (2.6) with the Schwarz operator $Q = Q_0 + Q_{11} + \ldots + Q_{JN_J}$.

Here for $i = 1, \ldots, J$; $j = 1, \ldots, N_i$ and for any $w \in S_h(\Gamma)$, $Q_{ij}w \in S_1^0(\Gamma_{ij})$ is the solution of the boundary element equation

$$\langle B_1 Q_{ij} w, v_{ij} \rangle = \langle B_1 w, v_{ij} \rangle \quad \forall v_{ij} \in S_1^0(\Gamma_{ij}) \tag{2.10}$$

and $Q_0 w \in S_H(\Gamma)$ solves

$$\langle B_1 Q_0 w, v_0 \rangle = \langle B_1 w, v_0 \rangle \quad \forall v_0 \in S_H(\Gamma). \tag{2.11}$$

In [MS97] we show for H_0 sufficiently small and $H_i \leq H_0$ that C_0, C_1 in (2.8) satisfy

$$C_0^{-1} \sim \max_{1 \leq i \leq J} \left(1 + \log \frac{H_i}{h_i}\right) \text{ and } C_1 = \text{const.} \tag{2.12}$$

3 Preconditioners for the Weakly Singular Operator

As subspace X_N^{-1} we use the space $S_h^0(\Gamma)$ of piecewise constant functions on a quasiuniform mesh of Γ. Let ϕ_j^h, $j = 1, \ldots, N-1$ denote the hat function in Section 2 and let the Haar basis function χ_j^h be defined as the derivative of ϕ_j^h. These functions χ_j^h together with the constant function 1 form a basis for $\bar{S} = S_h^0(\Gamma)$. We then decompose $\bar{S}$ as

$$\bar{S} = \bar{S}_0 + \bar{S}_1 + \ldots + \bar{S}_{N-1} \tag{3.13}$$

where $\bar{S}_0$ is defined as $S_h^0(\Gamma)$ with mesh size $H = 2h$ and where $\bar{S}_j = \text{span}\{\chi_j^h\}$, for $j = 1, \ldots, N-1$. Then an associate additive Schwarz method can be defined for the weakly singular integral equation via operators $\bar{Q}_j$ which are given by (2.5) with B_{-1} instead B_1 and $\bar{S}, \bar{S}_j$ substituting S, S_j respectively. Analogously to (2.9) a non-overlapping subspace decomposition of $\bar{S}$ may be introduced and corresponding operators $\bar{Q}, \bar{Q}_0, \bar{Q}_{ij}$ when using B_{-1} instead of B_1 in (2.11).

Then again for (3.13) the additive Schwarz method yields a GMRES method with convergence rates strictly less than 1 independently of the degrees of freedom [ST97b] whereas in the non-overlapping case again the constants C_0, C_1 of (2.8) show the behavior (2.12).

4 Numerical Results

The numerical experiments for the hypersingular integral equation (1.1) with $g_1(x) \equiv 1$ and wavenumber $k = 2.0$ on a quasiuniform mesh were performed on a SUN-Sparcstation 4/470 at the Institute for Applied Mathematics at University of Hannover. Here we give the eigenvalues and condition numbers which are linked to the rate of convergence by $C_1 = \lambda_{\max}$ and $C_0 = \sqrt{\lambda_{\min}}$. Table 1 gives the absolute values $\lambda_{\min}$, $\lambda_{\max}$ and the condition number of the unpreconditioned Galerkin system (1.3) and of the additive Schwarz preconditioner (2.6) with $H = 2h$.

In Table 2 the condition numbers for the additive Schwarz operator defined by (2.10) and (2.11) are given for different quotients H/h.

Table 1 Hypersingular integral equation (1.1) with $g_1(x) \equiv 1$: quasiuniform h-version, wavenumber $k = 2.0$

	Stiffness matrix			2-level		
N	$\lambda_{\min}$	$\lambda_{\max}$	cond	$\lambda_{\min}$	$\lambda_{\max}$	cond
31	8.8633d-02	1.1308	12.7586	1.0045	2.2324	2.2224
63	4.5142d-02	1.1333	25.1050	0.9923	2.2312	2.2484
127	2.2750d-02	1.1339	49.8413	0.9880	2.2304	2.2573
255	1.1399d-02	1.1340	99.4872	0.9875	2.2302	2.2582
511	5.7018d-03	1.1341	198.9038	0.9884	2.2300	2.2560
1023	2.8509d-03	1.1341	397.8011	0.9896	2.2299	2.2532

Table 2 Condition numbers for additive Schwarz operator of the hypersingular integral equation (1.1) with $g_1(x) \equiv 1$: quasiuniform h-version, wavenumber $k = 2.0$

$N \setminus H/h$	2	4	8	16	32	64	128
64	3.7124	3.9360	5.0803	5.9118	6.3642		
128	3.5003	4.0075	5.1320	6.1031	6.7030	7.0320	
256	3.3507	4.0258	5.2133	6.4471	7.5519	8.2994	8.3348
512	3.2436	4.0305	5.2341	6.5358	7.9045	9.1635	10.0478

5 Conclusion

The numerical examples clearly underline the theoretical results, i.e. the condition numbers of both methods are independent of the number of degrees of freedom but the condition numbers for the additive Schwarz operator defined by (2.10) and (2.11) depend logarithmically on H/h. Whereas the first method is only of theoretical interest due to the large coarse grid space, the second method can be implemented in an efficient and parallel way if we choose the size of the subspaces appropriately.

REFERENCES

[CW92] Cai X.-C. and Widlund O. B. (1992) Domain decomposition algorithms for indefinite elliptic problems. *SIAM J. Sci. Stat. Comput.* 13: 243–258.

[HS96] Hahne M. and Stephan E. P. (1996) Schwarz iterations for the efficient solution of screen problems with boundary elements. *Computing* 56: 61–85.

[LM72] Lions J. and Magenes E. (1972) *Non-Homogeneous Boundary Value Problems and Applications I.* Springer-Verlag, Berlin.

[MS97] Maischak M. and Stephan E. P. (1997) Two level methods for indefinite hypersingular integral equations – the h-version. (to appear).

[MST97] Maischak M., Stephan E. P., and Tran T. (1997) Domain decomposition methods for boundary integral equations of the first kind: numerical results. *Appl.Anal.* (to appear).

[ST97a] Stephan E. P. and Tran T. (1997) Domain decomposition algorithms for indefinite hypersingular integral equations. The h and p-versions. *SIAM J. Sci. Comput.* (to appear).

[ST97b] Stephan E. P. and Tran T. (1997) Domain decomposition algorithms for indefinite weakly singular integral equations. The h and p-versions. (to appear).

[Ste96] Stephan E. P. (1996) Additive Schwarz methods for integral equations of the first kind. In Whiteman J. (ed) *The Mathematics of Finite Elements and Applications IX, MAFELAP 1996.* Academic Press.

[SW84] Stephan E. P. and Wendland W. (1984) An augmented Galerkin procedure for the boundary integral method applied to two-dimensional screen and crack problems. *Appl. Anal.* 18: 183–219.

[SW90] Stephan E. P. and Wendland W. L. (1990) A hypersingular boundary integral method for two-dimensional screen and crack problems. *Arch. Rational Mech. Anal.* 112: 363–390.

[TS97] Tran T. and Stephan E. P. (1997) Preconditioners for the $h - p$ version of the Galerkin boundary element method. (to appear).

61

A space decomposition method for minimization problems

Xue–Cheng Tai and Magne Espedal

1 A Space Decomposition Algorithm

We consider:

$$\min_{v \in V} F(v) \ , \tag{1.1}$$

where functional F is differentiable and convex and space V is a reflexive Banach space. Our intention is to use space decomposition method to get some parallel domain decomposition and multigrid type algorithms for linear partial differential equations of the type

$$\left\{ \begin{array}{c} -\nabla \cdot (a \nabla u) = f \text{ in } \Omega \subset R^2 \ , \\ u = 0 \text{ on } \partial\Omega \ , \end{array} \right. \tag{1.2}$$

and for nonlinear elliptic problems like

$$\left\{ \begin{array}{ll} - & \nabla \cdot (|\nabla u|^{s-2} \nabla u) = f \text{ in } \Omega \subset \mathrm{R}^2 \ (1 < \mathrm{s} < \infty) \ , \\ & u = 0 \text{ on } \partial\Omega \ . \end{array} \right. \tag{1.3}$$

The algorithm given in this work were first proposed in [Tai92], see also [Tai94b], [Tai95a] and [Tai95b]. As the algorithm is proposed for a minimization problem, it is applicable for a wide class of problems, for example, eigenvalue problems, optimal control problems related to partial differential equations and least-squares method associated with linear and nonlinear equations.

A space decomposition method refers to methods that decompose the space V into a sum of subspaces, i.e. there are spaces $V_i, \ i = 0, 1, \cdots, m$ such that

$$V = V_0 + V_1 + \cdots + V_m \ . \tag{1.4}$$

For the decomposed spaces, we assume that there is a constant $C_1 > 0$ such that $\forall v \in V$, we can find $v_i \in V_i$ to satisfy:

$$v = \sum_{i=0}^{m} v_i \ , \qquad \text{and} \qquad \sum_{i=0}^{m} \|v_i\|_V^2 \le C_1^2 \|v\|_V^2 \ . \tag{1.5}$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Moreover, assume that there is a $C_2 > 0$ such that there holds

$$\sum_{i=0}^{m}\sum_{j=0}^{m}\langle F''(w_{ij})u_i, v_j\rangle \le C_2\left(\sum_{i=0}^{m}\|u_i\|_V^2\right)^{\frac{1}{2}}\left(\sum_{i=0}^{m}\|v_i\|_V^2\right)^{\frac{1}{2}} , \qquad (1.6)$$
$$\forall w_{ij} \in V, \forall u_i \in V_i, \forall v_j \in V_j \ .$$

Domain decomposition methods, multilevel methods and multigrid methods can be viewed as different ways of decomposing finite element spaces into sums of subspaces. For the estimation of the constants C_1 and C_2 for different types of decomposition of finite element methods for linear problems, one can find the proofs or references in Xu [Xu92]. If the space can be decomposed as in (1.4), then the following algorithm can be used to solve (1.1).

Algorithm 1 *(A multiplicative space decomposition method).*

1. *Choose initial values* $u_i^0 \in V_i$*.*
2. *For* $n \ge 1$*, find* $u_i^{n+1} \in V_i$ *sequentially for* $i = 0, 1, \cdots, m$ *such that*

$$F\left(\sum_{k<i}^{m} u_k^{n+1} + u_i^{n+1} + \sum_{k>i}^{m} u_k^n\right) \le F\left(\sum_{k<i}^{m} u_k^{n+1} + v_i + \sum_{k>i}^{m} u_k^n\right) , \quad \forall v_i \in V_i \ . \qquad (1.7)$$

3. *Go to the next iteration.*

In the following, we denote $u^n = \sum_{i=0}^{m} u_i^n$, $\forall n > 0$. By assuming that F is continuously differentiable and

$$K\|w - v\|_V^2 \le \langle F'(w) - F'(v), w - v\rangle \le L\|w - v\|_V^2 , \quad \forall w, v \in V , \qquad (1.8)$$

where $K > 0,\ L > 0$, and using $e^n = |\langle F'(u^n) - F'(u), u^n - u\rangle|^{\frac{1}{2}}$, as a measure of the error between u^n and u, the following convergence theorem is proved in Tai and Espedal [TE96].

Theorem 1 *If the space decomposition satisfies (1.5) and the functional F satisfies (1.8), then for Algorithm 1 we have:*

1. *If F is quadratic with respect to v and the norm of V is taken as* $\|v\|_V = \langle F'(v), v\rangle$*, then*

$$|e^{n+1}|^2 \le \frac{C_s^2}{1 + C_s^2}|e^n|^2 , \quad \forall n \ge 1 \ .$$

Above and also later, $C_s = C_2 C_1$*.*

2. *If F is third order continuously differentiable, then*

$$|e^{n+1}| \to 0 \text{ as n} \to \infty \ , \text{ and } \quad |\mathrm{e}^{\mathrm{n}+1}|^2 \le \beta_{\mathrm{n}}|\mathrm{e}^{\mathrm{n}}|^2 , \quad \forall \mathrm{n} \ge 1 ,$$

and the error reduction factor β_n *satisfies* $\lim_{n\to\infty}\beta_n = \frac{\frac{C_s{}^2}{K^2}}{1+\frac{C_s{}^2}{K^2}} < 1$*, which means the asymptotic convergence rate only depends on* C_s *and* K*.*

2 Application of the Space Decomposition to a Two-level Domain Decomposition Method

We use the space decomposition Algorithm 1 for a two-level overlapping domain decomposition method. For a given domain Ω, we first divide it into coarse mesh subdomains, and then refine each coarse mesh subdomain to get fine mesh divisions for Ω. In the following examples, domain Ω is taken as $[0,1] \times [0,1]$. Uniform mesh is used both for the coarse mesh division and the fine mesh division. Let $\Omega_i, i = 1, 2, \cdots$ be a coarse mesh division of Ω, see Figure 1, we then enlarge each Ω_i to $\Omega_i^\delta = \{T \in \mathcal{T}_h, dist(T, \Omega_i) \leq \delta\}$ to get overlapping subdomains. Here $\{\mathcal{T}_h\}$ denotes the fine mesh division for Ω.

The union of Ω_i^δ covers $\bar{\Omega}$ with overlaps of size 2δ. Let us denote the piecewise linear finite element space with zero traces on the boundaries $\partial\Omega_i^\delta$ as $S_0^h(\Omega_i^\delta)$, and denote S_h^0, S_H^0 as the coarse and fine mesh finite element spaces respectively. One can show that

$$S_0^h = S_0^H + \sum S_0^h(\Omega_i^\delta) \ . \tag{2.9}$$

For the overlapping subdomains, assume that there are m colors such that each subdomain Ω_i^δ can be marked with one color, and the subdomains with the same color will not intersect with each other. For suitable overlaps, we have $m = 4$, see Figure 1. Let Ω_i' be the union of the subdomains of the i^{th} color, and $V_i = \{v \in S_0^h \big| \ \ v(x) = 0 \text{ if } \ \ \mathrm{x} \notin \Omega_\mathrm{i}'\}$. By denoting subspaces $V_0 = S_0^H$ and $V = S_0^h$, we find that decomposition (2.9) means

$$V = V_0 + \sum_{i=1}^{4} V_i \ , \tag{2.10}$$

and so the two-level method is a way to decompose the finite element space. Moreover, let $V = H_0^1(\Omega)$ and F be the corresponding energy function of linear equation (1.2), then the constants in (1.5) and (1.6) are:

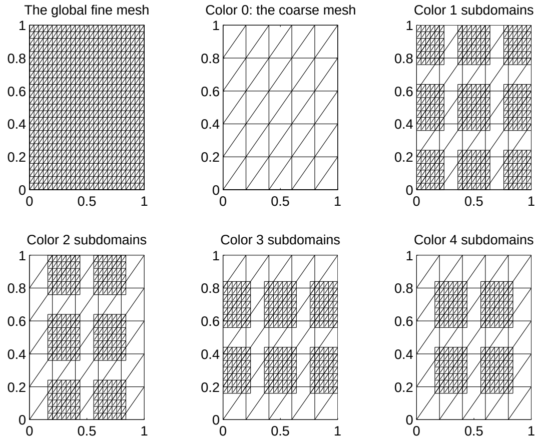
$$C_1 = C\sqrt{1 + \frac{H^2}{\delta^2}}, \ C_2 = Cm. \tag{2.11}$$

The proof for (2.11) can be found in different places, we refer to page 608 of Xu [Xu92] and Tai and Espedal [TE96]. By requiring $\delta = c_0 H$, where c_0 is a given constant, we have that C_1 and C_2 are independent of the mesh parameters h and H, and the number of the subdomains. So if the proposed algorithm is used, its error reduction per step does not depend on h and H.

3 Applications to Linear Elliptic Problems

We apply Algorithm 1 to solving linear problem (1.2). As was shown above, the two-level method is a space decomposition method. With the coarse mesh, the number of the subspaces is $m = 5$. For Algorithm 1, we define $w_i^{n+1} = \sum_{k<i}^{m} u_k^{n+1} + u_i^{n+1} +$

Figure 1 The coloring of the subdomains and the coarse mesh grid



$\sum_{k>i}^m u_k^n$ and $w_{-1}^{n+1} = u^n$. In each subdomain of the $i^{\rm th}$ color, the subproblem needs to be solved is

$$\begin{cases} (a\nabla w_i^{n+1}, \nabla v_i) = (f, v_i), & \forall v_i \in S_0^h(\Omega_i^\delta)\ , \\ w_i^{n+1} = w_{i-1}^{n+1} \text{ on } \partial\Omega_{\rm i}^\delta\ , & \end{cases} \tag{3.12}$$

and $w_i^{n+1} = w_{i-1}^{n+1}$ in $\Omega\backslash\Omega_{\rm i}'$. For the coarse mesh problem, if we let $w_H^{n+1} = u_0^{n+1} - u_0^n$, then it satisfies

$$(a\nabla(u^n + w_H^{n+\frac{1}{2}}), \nabla v_H) = (f, v_H)\ , \quad \forall v_H \in S_0^H(\Omega)\ . \tag{3.13}$$

After the computation of the subdomain problems and the coarse mesh problem, we set $u^{n+1} = w_m^{n+1}$. Note that the subdomains with the same color do not intersect with each other, so in computing the $i^{\rm th}$ color subdomain solutions, the computation is done in parallel in each of the subdomains of the $i^{\rm th}$ color. One observes that this is the standard multiplicative Schwarz method for linear elliptic equations. In the literature, this method is often symmetrised and then accelerated by conjugate gradient method, see [SBG96]. In the next example, we try to see the convergence without using the extra acceleration.

Example 1 *In this example, Algorithm 1 is tested for the case that* $a = e^{xy}, u = \sin(3\pi x)\sin(3\pi y)$*. For a given* N*, the coarse mesh size is taken as* $H = Hx = Hy = \frac{1}{N}$*. The fine mesh is then taken as* $h = hx = hy = \frac{1}{N^2}$*. Each subdomain is extended by* M *elements to get overlaps. The initial guess is taken as the coarse mesh solution* u_H*. Figure 2 illustrates the computed solution and computed error function. Table 1 shows the convergence property. For different tests with* $hx, hy \in [\frac{1}{125}, 1]$*, i.e. with unknowns* ≤ 15625*, and with overlap size* $\delta \approx \frac{H}{5}$*, the computed solution always converges to the global finite element solution in less than* 8 *steps.*

4 Applications to Linear Interface Problems

Figure 2 The computed solution with H=1/10, h=1/100, M=2.

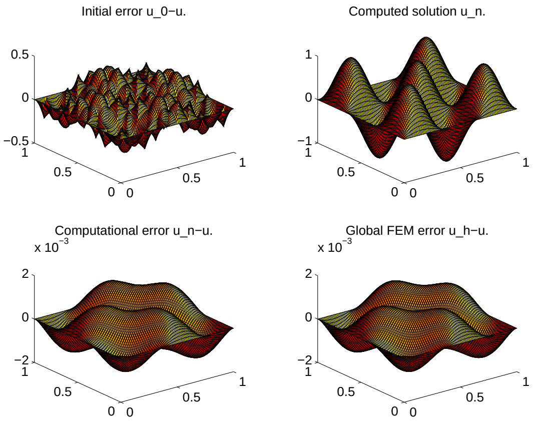
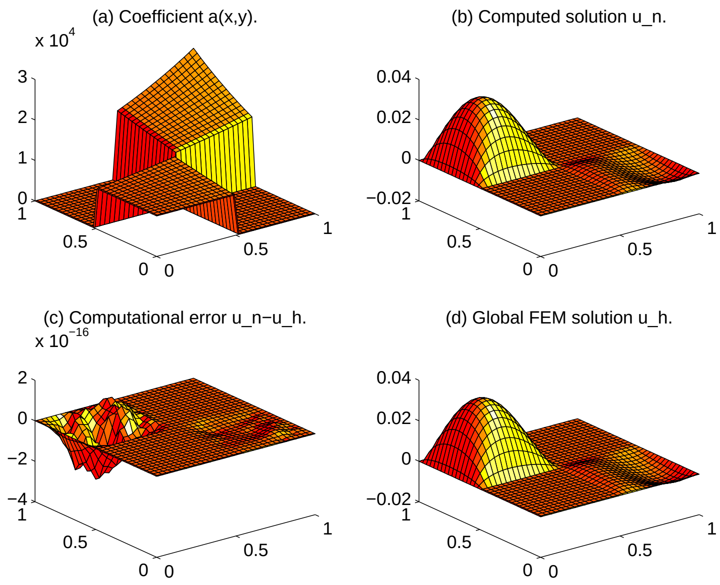


Table 1 *Maximum error with H=1/10, h=1/100, M=2.*

Iteration	*max-error*	*reduction*
1	*0.0729*	*0.24*
2	*0.0166*	*0.23*
3	*0.0036*	*0.22*
4	*0.0017*	*0.46*
5	*0.0015*	*0.88*

Figure 3 Computational results for a linear interface problem.



(a) Coefficient a(x,y).

(b) Computed solution u_n.

(c) Computational error u_n−u_h.

(d) Global FEM solution u_h.

Example 2 *We solve a linear interface problem. The coefficients are taken as* $a = c(x)e^{xy}$ *where* $c(x)$ *is piecewise constant and* $c(x) = 1$ *or* 10^4, *(see Figure 3). The global fine finite element solution is first computed. After that, the problem is computed by Algorithm 1 and the error between the iterative domain decomposition solution and the global fine mesh solution is calculated. The mesh sizes are* $hx = hy = \frac{1}{100}$, $Hx = Hy = \frac{1}{10}$. *The algorithm converges for arbitrary initial guesses. Each subdomain is extended by 2 elements to get overlap. The convergence is similar as the smooth problem.*

5 Applications to Nonlinear Elliptic Problems

In the literature, domain decomposition methods and multilevel methods have been intensively studied for linear elliptic problems. For nonlinear problems, it is hard to get some general convergence estimates. For literature results related to nonlinear problems, see [CD94], [CGKT94], [LSL89], [MX95], [Tai92]–[Tai94a], [Xu94], etc. The proposed algorithm of this work can be applied to linear problems (1.2) as well as nonlinear problems (1.3).

The Gauss-Newton method (Matlab subroutine fiminu) is used to solve the minimisation problem (1.7). Without using the domain decomposition, the original problem is simply too large and costly to be solved.

Example 3 *We use an analytical solution* $u = \sin(2\pi x)\sin(2\pi y)$ *for (1.3) to test Algorithm 1. Figure 4 and Table 2 show the computational results with fine mesh* $hx = hy = \frac{1}{100}$, *and coarse mesh* $Hx = Hy = \frac{1}{10}$. *Each subdomain is extended by 2*

Figure 4 The computational results for the nonlinear problem by Algorithm 1.

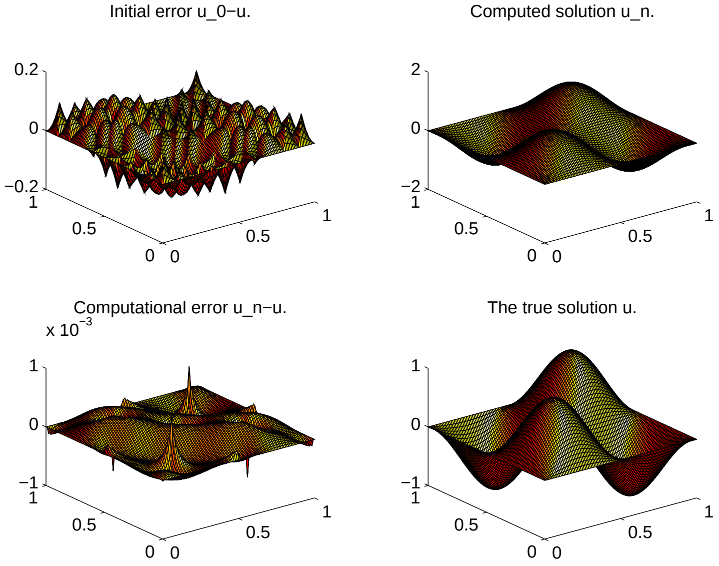


Table 2 Maximum error for the nonlinear problem by Algorithm 1.

Iteration	*max-error*	*reduction*
1	*0.1345*	
2	*0.0257*	*0.19*
3	*0.0049*	*0.19*
4	*0.0012*	*0.24*
5	*0.0007*	*0.60*
6	*0.0007*	*1.05*

elements to get overlaps. The initial guess is the coarse mesh solution. The value of s is 3. Numerical tests show that the algorithm converges for arbitrary initial guess and the error reduction does not depend on the initial guess. The dependency of the convergence on the overlapping size and on the number of subdomains is the same as for the linear problem (1.2).

Acknowledgement

This work is supported by VISTA, a research cooperation between the Norwegian Academy of Science and Letters and Den norske oljeselskap a.s. (Statoil) and the Research Council of Norway, under contract SEP-115837/431. The authors would like to thank Prof. J. Xu and Prof. P. Bjørstad for valuable discussions related to this

work.

REFERENCES

[CD94] Cai X.-C. and Dryja M. (1994) Domain decomposition methods for monotone nonlinear elliptic problems. In Keyes D. E. and Xu J. (eds) *Domain decomposition methods in scientific and engineering computing*, pages 21–28. American Mathematical Society, Providence.

[CGKT94] Cai X.-C., Groop W. D., Keyes D. E., and Tidriri M. D. (1994) Parallel implicit methods for aerodynamics. In Keyes D. E. and Xu J. (eds) *Domain decomposition methods in scientific and engineering computing*, pages 465–470. American Mathematical Society, Providence.

[LSL89] Lu T., Shih T. M., and Liem C. B. (1989) Parallel algorithm for variational inequalities based on domain decomposition. *Research report, IMS-33* .

[MX95] Marion M. and Xu J. (1995) Error estimates on a new nonlinear galerkin method based on two grid finite element. *SIAM J. Numer. Anal.* 32: 1170–1184.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. D. (1996) *Domain decomposition: parallel multilevel algorithms for elliptic partial differential equations.* Cambridge.

[Tai92] Tai X.-C. (1992) Parallel function decomposition and space decomposition methods with applications to optimisation, splitting and domain decomposition. *Preprint No. 231–1992, Institut für Mathematik, Technische Universität Graz* http://www.mi.uib.no/˜tai.

[Tai94a] Tai X.-C. (1994) Domain decomposition for linear and nonlinear elliptic problems via function or space decomposition. In Keyes D. and Xu J. (eds) *Domain decomposition methods in scientific and engineering computing (Proc. of the 7th international conference on domain decomposition, Penn. State University, 1993)*, pages 355–360. American Mathematical Society.

[Tai94b] Tai X.-C. (1994) Parallel function and space decomposition methods. In Neittaanmäki P. (ed) *Finite element methods, fifty years of the courant element*, Lecture notes in pure and applied mathematics, vol. 164, pages 421–432. Marcel Dekker inc.

[Tai95a] Tai X.-C. (1995) Parallel function and space decomposition methods – Part I. function decomposition. *Beijing Mathematics* 1, part 2: 104–134. http://www.mi.uib.no/˜tai.

[Tai95b] Tai X.-C. (1995) Parallel function and space decomposition methods – Part II. space decomposition. *Beijing Mathematics* 1, part 2: 135–152. http://www.mi.uib.no/˜tai.

[TE96] Tai X.-C. and Espedal M. (1996) Rate of convergence of a space decomposition method and applications to linear and nonlinear elliptic problems. Technical Report 103, Department of Mathematics, University of Bergen, Norway. http://www.mi.uib.no/˜tai.

[Xu92] Xu J. (1992) Iteration methods by space decomposition and subspace correction. *SIAM Rev.* 34: 581–613.

[Xu94] Xu J. (1994) A novel two-grid method for semilinear elliptic equations. *SIAM J. Sci. Comput.* 15: 231–237.

62

Nonoverlapping Domain Decomposition Methods for Inverse Problems

Karl Kunisch and Xue-Cheng Tai

1 Introduction

Inverse problems related to the estimation of coefficients of partial differential equations are ill-posed. Practical applications often use the fit-to-data output-least-squares method to recover the coefficients. In this work, we develop parallel nonoverlapping domain decomposition algorithms to estimate the diffusion coefficient associated with elliptic differential equations. In order to realize the domain decomposition methods, we combine the function decomposition approach of [Tai95a] and the augmented Lagrangian techniques of [IK90, KT97b]. The output-least-squares method minimizes the output error over the whole domain. When decomposing the domain into nonoverlapping subdomains the output error over the whole domain equals the sum of the output errors in the subdomains. Thus, by borrowing ideas from [Tai95a], parallel methods can be used to find the minimizer. In this approach the partial differential equation arises as a constraint in the optimization problem whose proper treatment is essential. We incorporate it by an augmented Lagrangian technique.

To present the approach we consider the model problem

$$\left\{\begin{array}{rcll} -\nabla\cdot(q\nabla u) & = & f & \text{in} \quad \Omega \\ u & = & 0 & \text{on} \quad \partial\Omega \end{array}\right. \tag{1.1}$$

where Ω is a domain with boundary $\partial\Omega$, $q \in L^\infty(\Omega)$ and $q \geq \alpha > 0$ and $f \in L^2(\Omega)$ is a given function. The problem consists in estimating the functional parameter q from an observation u_d of the state variable u. The idea that will be described for this model problem can be extended to other parameter estimation problems of partial differential equations.

The fit-to-data formulation for the above estimation problem is given by

$$(\mathcal{P}) \qquad \left\{\begin{array}{rl} \min & \frac{\beta}{2}|q-q_d|_N^2 + \frac{1}{2}|u-u_d|_{H^1}^2 \\ \text{subject to} & (q,u) \text{ satisfying (1.1)}\ . \end{array}\right.$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Here $|\cdot|_N$ denotes a norm or seminorm on $H^2(\Omega), q_d$ is an initial guess for the parameter and $\beta \geq 0$ stands for the regularization parameter. Thus $(\mathcal{P})$ represents a regularized least-squares formulation with the partial differential equation as a constraint. When decomposing the domain Ω into nonoverlapping subdomains the output error in the whole domain Ω equals the sum of the output errors in the subdomains. Thus, by borrowing ideas from [Tai95a, Tai95b, Tai94], parallel methods can be used to find the minimizer. However, the partial differential equation which arises as a constraint in $(\mathcal{P})$ represents an essential difficulty. This constraint will be incorporated by an augmented Lagrangian technique of [IK90],[KT97b]. Domain decomposition methods for solving the state equation (1.1) for given q and f have been extensively studied. There is a vast literature of which we only mention some relevant classical papers [BPS86], [BW86] and [MQ89].

2 The Domain Decomposition Approach

Through out this work Ω is assumed to be a two-dimensional, bounded, simply connected convex domain with piecewise smooth boundary. We decompose Ω into finitely many nonoverlapping subdomains. The decomposition is carried out in such a way that all subdomains are marked with two colours, say white and black. Subdomains do not intersect each other and the union of their closures equals the closure of Ω. Moreover subdomains with the same colour do not meet each other along edges but rather only at most at one corner. We denote by Ω_1 and Ω_2 the union of the white and black subdomains respectively. Let $\Gamma_1 = \partial\Omega_1$ and $\Gamma_2 = \partial\Omega_2$. Then the interfaces between the subdomains are $\Gamma = \Gamma_1 \cap \Gamma_2 = \Gamma_1 \setminus \partial\Omega = \Gamma_2 \setminus \partial\Omega$. We shall utilise the following notation:

$$\begin{aligned}
V_i &= \{v | v \in H^1(\Omega_i), \quad v = 0 \text{ on } \Gamma_i \cap \partial\Omega\},\\
W_i &= \{v | v \in H^2(\Omega_i)\},\\
K_i &= \{v | v \in H^2(\Omega_i), \quad v \geq \alpha > 0 \text{ a.e. in } \Omega_i\}, \text{ for } i = 1, 2,
\end{aligned}$$

$$\begin{array}{rclcl}
X_1 &=& W_1 \times W_2 &=& \{v | v_{|\Omega_i} \in H^2(\Omega_i), \quad i = 1, 2\},\\
X_2 &=& V_1 \times V_2 &=& \{v | v_{|\Omega_i} \in H^1(\Omega_i), \quad i = 1, 2, v = 0 \text{ on } \partial\Omega\},\\
K &=& K_1 \times K_2 &=& \{v | v_{|\Omega_i} \in H^2(\Omega_i), \quad i = 1, 2, v \geq \alpha > 0 \text{ a.e. in } \Omega\},\\
X &=& X_1 \times X_2. &&
\end{array}$$

Except for K_i and K the above sets represent Hilbert spaces with their conventional inner products and norms. Each of the subdomains Ω_1 and Ω_2 may consist of some disconnected components. Thus W_1, for example, can equivalently be expressed as $\prod_{j=1}^m H^2(\Omega_{1j})$, where m denotes the number of disconnected components Ω_{1j} of Ω_1. The functions from X_1 and X_2 have jumps along the interfaces of Ω_1 and Ω_2.

For $x \in X$ we shall use the notation $x = (q_1, q_2, u_1, u_2) = (q, u)$. The function-space formulation of $(\mathcal{P})$ is given by

$$\text{(P)} \qquad \left\{ \begin{array}{rl} \min & \frac{\beta}{2}|q - q_d|_N^2 + \frac{1}{2}|u - u_d|_{H^1}^2 \\ \text{subject to} & (q, u) \in K \times H_0^1(\Omega) \text{ satisfying (1.1).} \end{array} \right.$$

Here (q,u) is called solution to (1.1) if

$$(q\nabla u, \nabla v)_\Omega = (f,v) \text{ for all } v \in H_0^1(\Omega).$$

Using the fact that $q \in L^\infty(\Omega)$ provided that $q \in K$, it is simple to argue the existence of a solution $x^* = (q^*, u^*)$ to (P). Next we define the mappings

$$e_1 : X \to V_1, \qquad e_2 : X \to V_2,$$

such that for $x \in X, e_1(x) \in V_1$ and $e_2(x) \in V_2$ are the solutions to the following problems (2.2) and (2.4):

$$\left\{ \begin{array}{ll} (\nabla e_1, \nabla v_1)_{\Omega_1} + (e_1, v_1)_{\Omega_1} & = (q_1 \nabla u_1, \nabla v_1)_{\Omega_1} - (f, v_1)_{\Omega_1} \\ & \quad \text{for all } v_1 \in H_0^1(\Omega_1) \ , \\ & e_1 = u_1 - u_2 \text{ on } \Gamma \ , \\ & e_1 = 0 \text{ on } \partial\Omega \cap \Gamma_1 \ . \end{array} \right. \tag{2.2}$$

For any $v \in V_2$, let Rv be an extension to Ω_1 satisfying $Rv = 0$ on $\partial\Omega \cap \Gamma_1, Rv = v$ on Γ, and

$$\|Rv\|_{H^1(\Omega_1)} \leq C_1 \|v\|_{H^1(\Omega_2)}, \tag{2.3}$$

with constant C_1 independent of $v \in V_2$. Note that the harmonic extension operator R_H defined by $R_H v = 0$ on $\partial\Omega \cap \Gamma_1$, $R_H v = v$ on Γ and

$$(\nabla R_H v, \nabla \phi)_{\Omega_1} = 0 \ , \quad \forall \phi \in H_0^1(\Omega_1)$$

satisfies the required properties. For numerical purposes we prefer to use a different extension which will be introduced in section 5. With R thus defined, let $e_2 \in V_2$ be the solution to

$$\left\{ \begin{array}{l} (\nabla e_2, \nabla v_2)_{\Omega_2} + (e_2, v_2)_{\Omega_2} = (q_2 \nabla u_2, \nabla v_2)_{\Omega_2} - (f, v_2)_{\Omega_2} \\ \quad + \langle u_2 - u_1, v \rangle_{\partial\Omega_2} + (q_1 \nabla u_1, \nabla R v_2)_{\Omega_1} - (f, Rv_2)_{\Omega_1} \text{ for all } v \in V_2, \\ e_2 = 0 \text{ on } \partial\Omega \cap \Gamma_2 \ . \end{array} \right. \tag{2.4}$$

With e_1 and e_2 defined we introduce $e : X \to X_2$ by

$$e(x) = (e_1(x), e_2(x)).$$

We choose $|\cdot|_N$ in $(\mathcal{P})$ to be the piecewise $H^2(\Omega)$ norm on Ω_1 and Ω_2 and define

$$\begin{array}{rcl} J_1(x) & = & \frac{\beta}{2}\|q_1 - q_d\|^2_{H^2(\Omega_1)} + \frac{1}{2}\|u_1 - u_d\|^2_{H^1(\Omega_1)} \\ J_2(x) & = & \frac{\beta}{2}\|q_2 - q_d\|^2_{H^2(\Omega_2)} + \frac{1}{2}\|u_2 - u_d\|^2_{H^1(\Omega_2)}, \\ J(x) & = & J_1(x) + J_2(x). \end{array}$$

We shall focus on the minimization problem

$$(PP) \qquad \min_{(q,u) \in K \times X_2, \ (q,u)=0.} J(x)$$

Its relation to (P) is established in the following lemma: (see [KT97a])

Lemma 2.1 $x^* = (q^*, u^*)$ *is a minimizer of (PP) if and only if it is a minimizer of (P).*

3 The Augmented Lagrangian Method

In this section we develop parallel nonoverlapping domain decomposition algorithms for (PP). In [Tai95a, Tai95b] the following problem was considered:

$$\min_{x\in K}\sum_{i=1}^{m} F_i(x), \qquad K\subset V.$$

Under the assumption that K is convex and closed in the Hilbert space V and that the functions F_i are convex or uniformly convex several parallel algorithms based on the augmented Lagrangian method were obtained. Problem (PP) does not fit into this class of problems since the constraints of (PP) are not convex. We shall therefore combine the ideas from [Tai95a, Tai95b] and the techniques from [IK90], [KT97b] to overcome this difficulty.

Let us review some of the results of [KT97b] and consider

$$\min_{e(q,u)=0,\ (q,u)\in K\times X_2} \frac{\beta}{2}\|q-q_d\|^2_{X_1}+\frac{1}{2}\|u-u_d\|^2_{X_2}\ . \tag{3.5}$$

In [KT97b], the constraint e is assumed to have the special structure

$$e(q,u)=b(q,u)+l_1(q)+l_2(u)+\tilde f,$$

with $b: X\to Y,\quad l_i\in\mathcal{L}(X_i,Y),\quad \tilde f\in Y,\quad (q_d,u_d)\in X,\quad X=X_1\times X_2$, and b a bounded bilinear form satisfying

$$\|b(q,u)\|_Y\le\|b\|\ \|q\|_{X_1}\ \|u\|_{X_2} \text{ for all } (q,u)\in X.$$

Here X_1, X_2 and Y are real Hilbert spaces. In the context of Section 2 the spaces X_i assume the specific meaning explained there and $Y=X_2$. For any $q\in X_1$ let $A_q\in\mathcal{L}(X_2,Y)$ denote the operator defined by

$$A_q v=b(q,v)+l_2(v).$$

Under the conditions

(H1) e is continuous from the weak topology on X to the weak topology on Y, (which guarantees the existence of a solution $x^*=(q^*,u^*)$ to (3.5)),
(H2) A_{q^*} is a homeomorphism from X_2 to Y,
(H3) $\|u^*-u_d\|_{X_2}\|(A^*_{q^*})^{-1}\|_{\mathcal{L}(Y,X_2)}\ \|b\|\le\sqrt{\beta}$,

the solution x^* to (3.5) is unique and the following

Algorithm 1
Step 1. *Choose* $\lambda_0, c>0, \sigma\in(0,c]$. *For* $n=1,2,\ldots$ *do:*
Step 2. *Determine* x^n *as the solution to*

(P_{aux}) $\qquad \min L_c(x,\lambda^{n-1})$ *over* $x\in K\times X_2$.

Step 3. *set* $\lambda^n=\lambda^{n-1}+\sigma e(x^n)$, produces a sequence $\{x^n\}$ such that $\lim_{n\to\infty} x^n = x^*$. If $q^* \in \operatorname{int} K$ and σ is chosen appropriately then x^n converges linearly in X to x^*. Moreover λ^n converges weekly in Y to λ^*, where λ^* is the Lagrange multiplier associated to the constraint $e(x) = 0$. In the above algorithm $L_c(x, \lambda)$ is the augmented Lagrangian functional

$$L_c(x,\lambda) = \frac{\beta}{2}\|q - q_d\|^2_{X_1} + \frac{1}{2}\|u - u_d\|^2_{X_2} + (\lambda, e(x))_Y + \frac{c}{2}\|e(x)\|^2_Y.$$

In [KT97a], it was shown that (PP) fits into the general framework of problem (3.5) and that (H1) - (H3) are satisfied.

4 Nonoverlapping Domain Decomposition for (P_{aux})

In this section we describe a Gauss-Seidel iteration to solve (P_{aux}). We utilise the decomposition of Ω into M white and M black subdomains as described at the beginning of Section 2 and use M parallel processors. Each processor takes care of a white and a neighbouring black subdomain.

Let us recall the augmented Lagrangian functional that appears as the cost functional in (P_{aux}):

$$\begin{aligned} L_c(x,\lambda) = J_1(x) + J_2(x) + (\lambda_1, e_1(x))_{V_1} + (\lambda_2, e_2(x))_{V_2} \\ +\tfrac{c}{2}\|e_1(x)\|^2_{V_1} + \tfrac{c}{2}\|e_2(x)\|^2_{V_2}, \end{aligned}$$

and $x = (q_1, q_2, u_1, u_2) = (q, u) \in X$. We note that J_i is only a function of $x_i = (q_i, u_i), i = 1, 2$. The coupling between x_1 and x_2 occurs through the boundary constraints described by e_1 and e_2. In the Gauss-Seidel algorithm $L_c(x, \lambda)$ is minimized with respect to x in the following order: $q_1 \to u_1 \to q_2 \to u_2$. The algorithm is:

Algorithm 2

(i) Choose $u^{n,0} \in X_2, q_2^{n,0} \in W_2$ if $n = 1$, else set $u^{n,0} = u^{n-1}, q_2^{n,0} = q_2^{n-1}$.
(ii) For $k = 1, 2 \ldots$ do: find $q_i^{n,k}, u_i^{n,k}$ such that
a) $L_c(q_1^{n,k}, u_1^{n,k-1}, q_2^{n,k-1}, u_2^{n,k-1}, \lambda^{n-1}) \le L_c(q_1, u_1^{n,k-1}, q_2^{n,k-1}, u_2^{n,k-1}, \lambda^{n-1})$, for all $q_1 \in K_1$.
b) $L_c(q_1^{n,k}, u_1^{n,k}, q_2^{n,k-1}, u_2^{n,k-1}, \lambda^{n-1}) \le L_c(q_1^{n,k}, u_1, q_2^{n,k-1}, u_2^{n,k-1}, \lambda^{n-1})$, for all $u_1 \in V_1$.
c) $L_c(q_1^{n,k}, u_1^{n,k}, q_2^{n,k}, u_2^{n,k-1}, \lambda^{n-1}) \le L_c(q_1^{n,k}, u_1^{n,k}, q_2, u_2^{n,k-1}, \lambda^{n-1})$, for all $q_2 \in K_2$.
d) $L_c(q_1^{n,k}, u_1^{n,k}, q_2^{n,k}, u_2^{n,k}, \lambda^{n-1}) \le L_c(q_1^{n,k}, u_1^{n,k}, q_2^{n,k}, u_2, \lambda^{n-1})$, for all $u_2 \in V_2$.

The sequences $\{u_i^{n,k}\}$ and $\{q_i^{n,k}\}$ converge to the solution of (P_{aux}), see [KT97a].

5 Numerical Tests.

Experiments for one and two dimensional problems were carried out. Uniform triangular mesh and linear finite element functions were used for 2D approximations.

Table 1 The numerical errors of the first 10 steps.

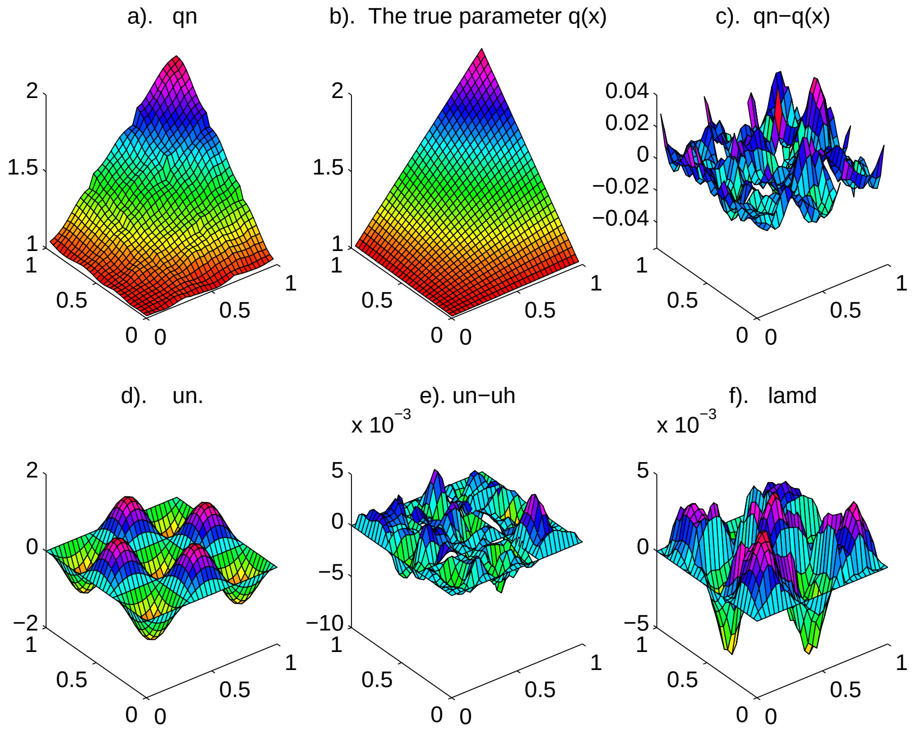
Iteration	err_q^*	err_q
1	0.03938	0.3363
2	0.02473	0.01972
3	0.0191	0.007926
4	0.01607	0.004409
5	0.01418	0.002879
6	0.01292	0.002071
7	0.01204	0.001589
8	0.01142	0.001273
9	0.01096	0.001055
10	0.01064	0.0008957

For a given q^* and f, the observation u_d is obtained by adding uniformly distributed random numbers in $[-\delta, \delta]$ to the finite element solution at the nodal points. In the simulations, Algorithm 2 is applied to find the minimizer of (P_{aux}) and only 3 iterations are performed between the subproblems a)–b)–c)–d) in our computations. A very simple extension operator is used in the computations by choosing $Rv = v$ in Ω_2 and $Rv(x_i) = 0$ if x_i is an inner node of Ω_1. Condition (2.3) is fulfilled with a constant C_1 depending on the mesh size h. Our one dimensional tests show that the convergence rate does not depend on the size of the extension. This may be due to the fact that the most important source for inaccurate reconstruction of q is the ill-posedness of the estimation problem.

From our numerical experiments, we find that there are two issues that require special care in using Algorithm 2. First, it must be guaranteed that the subproblems a) and c) of Algorithm 2 are identifiable when decomposing the domain. Sufficient conditions for the identifiability can be found in [IK94]. Second, identifying q_2 from c) of Algorithm 2 is very sensitive to observation errors and the errors caused by the initial values. One way to overcome such a problem is to choose the boundary conditions according to the flow directions. Dirichlet boundary conditions shall be used in the outflow boundaries and Neumann boundary conditions need to be used on the inflow boundaries. This is beyond the scope of the present work and shall be reported in detail in [KT97a].

Figure 1 depicts a typical numerical result. We identify $q(x, y) = e^{xy}$ from $u(x, y) = sin(3\pi x) sin(3\pi y)$. The domain $\Omega = (0, 1) \times (0, 1)$ is divided into $3 \times 3 = 9$ subdomains. Dirichlet boundary conditions are used for the subdomains at the 4 corners and the one in the middle. Extension operators are used for the other subdomains. Observation error is added with $\delta = 0.01$. In the computations, we use mesh size $h = 1/30$, $c = 100$, $\sigma = 100$, and $\beta = 0.1$. The convergence for the first 10 iterations are shown in Table 1. In the table, $err_q^* = \|q_1^n - q^*\|_{L^2(\Omega_1)} + \|q_2^n - q^*\|_{L^2(\Omega_2)}$ and $err_q = \|q_1^n - q_1^{n-1}\|_{L^2(\Omega_1)} + \|q_2^n - q_2^{n-1}\|_{L^2(\Omega_2)}$.

Figure 1 The identified parameter by domain decomposition.



Acknowledgement

The authors would like to thank Dr. J. Frøyen for carrying out some one dimensional tests and Professor M. Espedal for valuable discussions. The work of the second author was supported by the Research Council of Norway, under contract SEP-115837/431.

REFERENCES

[BPS86] Bramble J. H., Pasciak J. E., and Schatz A. H. (1986) An iterative method for elliptic problems on regions partitioned into substructures. *Math. Comp.* 46: 361–369.

[BW86] Bjørstad P. and Widlund O. (1986) Iterative methods for the solution of elliptic problems on regions portioned into substructuring. *SIAM J. Numer. Anal.* 23: 1097–1120.

[IK90] Ito K. and Kunisch K. (1990) The augmented lagrangian method for parameter estimation in elliptic systems. *SIAM J. Control Optim.* 28: 113–136.

[IK94] Ito K. and Kunisch K. (1994) On the injectivity and its linearization of the coefficient to solution mapping for elliptic boundary value problems. *J. Math. Anal. Appl.* 188: 1040–1066.

[KT97a] Kunisch K. and Tai X.-C. (1997) Domain decomposition methods for elliptic parameter estimation problems. *Preprint* .

[KT97b] Kunisch K. and Tai X.-C. (1997) Sequential and parallel splitting methods for bilinear control problems in Hilbert spaces. *SIAM J. Numer. Anal.* 43: 91–118.

[MQ89] Marini L. D. and Quarteroni A. (1989) A relaxation procedure for domain decomposition methods using finite elements. *Numer. Math.* 55: 575–598.

[Tai94] Tai X.-C. (1994) Parallel function and space decomposition methods. In Neittaanmäki P. (ed) *Finite element methods, fifty years of the courant element*, Lecture notes in pure and applied mathematics, vol. 164, pages 421–432. Marcel Dekker inc.

[Tai95a] Tai X.-C. (1995) Parallel function and space decomposition methods – part I. function decomposition. *Beijing Mathematics* 1, part 2: 104–134. http://www.mi.uib.no/~tai.

[Tai95b] Tai X.-C. (1995) Parallel function decomposition and space decomposition methods: Part II. space decomposition. *Beijing Mathematics* 1, part 2: 135–152. http://www.mi.uib.no/~tai.

63

Capacitance Matrix Preconditioning

Karina Terekhova

1 Introduction

Iterative methods, widely used for solutions of large linear systems require preconditioning as an essential part. The advent of parallel computers motivates a search for preconditioners suitable for parallel processing.

A domain decomposition approach can satisfy this demand. The domain of definition of the problem is partitioned into subdomains, and the original problem is substituted by an equivalent one, defined on the internal boundaries (interfaces) separating the subdomains. This smaller problem is solved by an iterative method, usually with the help of preconditioning to accelerate the convergence. A preconditioner in this case must be an easily invertible approximation to the interface operator, also called the capacitance matrix, or the Schur complement.

A good approximation to the Schur complement of a linear system can be constructed algebraically by investigating its numerical structure. This idea was introduced by M. Dryja [Dry82] and developed in a paper by G. Golub and D. Mayers [GM83] that referred to the symmetric 2D case. This paper shows how the underlying reasoning can be extended to design a similar preconditioner for other elliptic problems.

2 Problems in Two Dimensions

Let us consider a symmetric model problem $\Delta u = f$ defined on a rectangular region subdivided into two rectangular parts with homogeneous Dirichlet conditions imposed on the outer boundary. The Schur complement S is then symmetric and positive definite. To solve the equation for S efficiently we need the preconditioning matrix M to be close to S and especially for the eigenvalues of $M^{-1}S$ to be clustered as closely as possible.

Examination of the Schur complements in some particular cases shows that the elements of S are dependent mainly on the distance from the diagonal, $\mid i - j \mid$, with

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

the largest element on the diagonal, the elements decreasing quite rapidly as $| i - j |$ increases.

This suggests that a useful approximation to S may be found by letting the boundaries of the two subdomains move to infinity. Then the setting is: find the solutions of Laplace's equation in the two half-planes, the solution being required to vanish at infinity and also at all points on the dividing axis, except at the origin, where it is equal to one.

Denoting by r and s the Cartesian indices along and normal to the interface in a uniform two-dimensional grid we have

$$\begin{cases} u_{r,s-1} + u_{r,s+1} + u_{r+1,s} + u_{r-1,s} - 4\, u_{r,s} = 0 \\ u_{r,s} \to 0 \text{ as } r \to \pm\infty, s \to \infty \\ u_{r,0} = 0 \ (r \neq 0) \\ u_{0,0} = 1 \end{cases}$$

Defining the generating function

$$\phi_s(t) = \sum_{r=-\infty}^{\infty} t^r u_{r,s}$$

we obtain the solution from the characteristic equation

$$\phi_s(t) = \left[2 - \frac{1}{2}\left(t + \frac{1}{t}\right) - \left(\left\{2 - \frac{1}{2}\left(t + \frac{1}{t}\right)\right\}^2 - 1\right)^{1/2}\right]^s$$

The residuals at the grid points on the axis are given by

$$\rho_r = u_{r-1,0} + u_{r+1,0} + 2\, u_{r,1} - 4\, u_{r,0}$$

for which the generating function is

$$\psi(t) = \left(t + \frac{1}{t} - 4\right)\, \phi_0 + 2\, \phi_1 = -2\, \left\{\left[2 - \frac{1}{2}\left(t + \frac{1}{t}\right)\right]^2 - 1\right\}^{1/2}$$

We then expand ψ in positive and negative powers of t to obtain ρ_r which is the coefficient of t^r.

$$\begin{aligned} \rho_r &= \frac{1}{2\pi}\int_{-\pi}^{\pi} -2\cos r\theta\, [(2-\cos\theta)^2 - 1]^{1/2} d\theta \\ &= -\frac{4}{\pi}\int_{0}^{\pi} \cos 2k\alpha\ \sin\alpha\, [1 + \sin^2\alpha]^{1/2} d\alpha \end{aligned}$$

A possible preconditioner then is $M_{ij}^{(1)} = \rho_{|i-j|}$. Full details can be found in [GM83].

The described method can be applied with some changes to a convection-diffusion equation of the form

$$-\varepsilon\, \Delta\, u + a\, u_x + b\, u_y = f,$$

where ε, a and b are constants with $a \geq 0, b \geq 0, \varepsilon > 0$, and where ε may be small compared with a and b. As before, the problem is defined on a rectangular region subdivided into two rectangular parts. The Schur complement is, of course, unsymmetric, but the dependence of its elements on the distance from the diagonal is still quite clear. This justifies approximation of the problem with the boundaries of the region moving away to infinity, just as in the symmetric case. Upon upwind finite differencing the system to solve is

$$\begin{cases} u_{r,s+1} + u_{r+1,s} + A\, u_{r-1,s} + B\, u_{r,s-1} - C\, u_{r,s} = 0 \\ u_{r,s} \to 0 \text{ as } r \to \pm\infty, s \to \infty \\ u_{r,0} = 0 \ (r \neq 0) \\ u_{0,0} = 1 \end{cases}$$

where $A = 1 + ah/\varepsilon,\ B = 1 + bh/\varepsilon,\ C = 2 + A + B$.

Again, ρ_r is the coefficient of t^r and the residuals on the axis are finally

$$\rho_r = \frac{1}{2\pi} \int_{-\pi}^{\pi} \cos r\theta \left[\left(2\sqrt{A}\, \cos\theta - C \right)^2 - 4B \right]^{1/2} d\theta$$

so the preconditioner should be defined as

$$M_{ij} = A^{-\frac{i-j}{2}} \rho_{|i-j|}.$$

Note that the preconditioner is unsymmetric and its elements grow rapidly above the diagonal and decrease under it.

As the preconditioner contains an exponential quantity $A^{-\frac{i-j}{2}}$ which depends on the parameter a of the problem, evaluation of the solution far from the interface may cause floating point precision loss. A sufficient number of interfaces and a suitable choice of the grid size h should be used in order to avoid this problem.

3 Problems in Three Dimensions

Let us consider a model problem $\Delta u = f$ defined on a cube subdivided into two parts by a plane.

The reasoning for the two-dimensional model problem applies to the three-dimensional one if the relevant changes are made to the formulation of the discretised infinite problem. Thus, after the outer boundaries have moved to infinity, we have

$$\begin{cases} u_{r+1,s,t} + u_{r-1,s,t} + u_{r,s-1,t} + u_{r,s+1,t} + \\ u_{r,s,t+1} + u_{r,s,t-1} - 6\, u_{r,s,t} = 0 \\ u_{r,s,t} \to 0 \text{ as } r \to \pm\infty, s \to \pm\infty, t \to \infty \\ u_{r,s,0} = 0 \ (r, s \neq 0) \\ u_{0,0,0} = 1 \end{cases}$$

We solve the discretised problem for the generating functions, and then separate the desired residuals as coefficients of double Fourier series. The residuals obtained are

$$\rho_{r,s} = -\frac{8}{\pi^2} \int_0^{\pi} \int_0^{\pi} \cos r\alpha\, \cos s\beta\, ([3 - \cos\alpha - \cos\beta]^2 - 1)^{1/2} d\alpha d\beta$$

These residuals, ordered with respect to the ordering of variables in the original problem, form the preconditioner. It can be dense if all residuals are used, or it can take block-diagonal or banded forms if we substitute the residuals which are close to zero in some sense by zeroes. This does not usually cause the loss of convergence properties and gives the obvious advantage of easy inversion.

4 Results

In this section we discuss the practical aspects of applying the capacitance matrix preconditioner, serially and in parallel, to model and real-life test problems in two and three dimensions. All industrial examples were supplied by Elf Geoscience Research Centre. They were obtained from convection-diffusion equations modelling the process of oil recovery.

Methods

The particular iterative method used in the numerical experiments is BiCGSTAB, proposed by H. van der Vorst (see [vdV92]). In it two solves of the subproblems and two applications of the preconditioner are required per iteration. The preconditioner is calculated and inverted in advance, so its application is computationally cheap. The most time-consuming operation is, therefore, solving the subproblems. Exact solvers (Gaussian elimination), direct solvers (for example, fast Fourier transform solver) and various iterative techniques are proposed in the literature for this purpose.

We used the ORTHOMIN algorithm with the nested factorisation preconditioner as a solver for subproblems in three dimensions. ORTHOMIN is an optimal and minimal conjugate-gradient-like algorithm showing fast reliable convergence at the expense of relatively high storage requirements.

Many problems relevant for the industrial applications take block diagonal form after discretisation. This particular structure of their matrices can be exploited to achieve efficient solution of subproblems. The algorithm of nested factorisation preconditioning, although not easily adapted to deal with general sparse matrices, is particularly good for block tridiagonal ones. The algorithms of ORTHOMIN and of recursive evaluation of the nested factorisation preconditioner is given in [ACP81].

The nested factorisation preconditioner can hardly be parallelised without considerable loss of efficiency because of data interdependence. However, ORTHOMIN with the nested factorisation preconditioning make a fast and predictable serial solver of the 3D subproblems in a parallel iterative solution process preconditioned by the capacitance matrix preconditioner.

Parallel Model

The parallel program was written following the bulk-synchronous parallel (BSP) paradigm.

The BSP model, introduced by L. Valiant in 1990 ([Val89]), implements the idea of portable parallel software. A BSP computation consists of a number of asynchronous supersteps during which the processors can issue requests for non-local read or write

Table 1 2D: Nonsymmetric model problem

		Direct solver	Neumann preconditioner	New preconditioner
32×32	iterations	n/a	22	3
	time	75	126	105
	Mflops	2.4	7.9	4.0
64×64	iterations	n/a	39	3
	time	1259	1727	877
	Mflops	40.1	114.2	35.8

Table 2 3D: Nonsymmetric model problem of size $44 \times 17 \times 14$ solved serially

Subdomains	time	iterations
2	80	5
4	82	7

operations. Each superstep is followed by a synchronisation session which ensures that all information exchange is completed.

The total cost of a BSP computation can be expressed in terms of separate computation, communication and synchronisation costs, combined with the parameters of the computer reflecting its performance in computation, communication and synchronisation. Variation of these costs with the change in the number of processors is predictable, as well as the performance of a particular parallel computer running a given algorithm.

We have obtained the portable cost estimates for the capacitance matrix preconditioner in combination with BiCGSTAB. The computational cost of one iteration of the proposed algorithm is

$$\begin{array}{ll} \text{3D:} & O(\frac{n}{p} + gn^{2/3} + l) \\ \text{2D:} & O(\frac{n}{p} + gn^{1/2} + l) \end{array}$$

where n is the number of grid points, p is the number of processors, g characterises the communication throughput and l is the synchronisation latency of the parallel computer.

The cost of an iteration of unpreconditioned BiCGSTAB is of the same order of magnitude. This means that the capacitance matrix preconditioner increases the cost of a BiSGCTAB iteration only by a constant factor. The great reduction in the number of iterations justifies the small extra cost of preconditioning.

Test Cases

The preconditioner was tried on model and real-life examples, both in two and three dimensions. Two-dimensional examples were solved serially, examples in three

Table 3 3D: Model problem of size $46 \times 19 \times 5$ solved in parallel

Processors	Serial (time)	BSP (time)	iterations
2	59	31	3
4	61	18	3

dimensions serially and in parallel.

The serial programs were run on a SUN SPARC workstation; the parallel program was run on several workstations connected via a network. All programs were written in Fortran 77 using double precision arithmetic.

The model problems in two dimensions were derived from the convection-diffusion equations of the form described in Section 2. The coefficients a and b varied in the range $[0, 5]$ with ε a constant varying between 10^{-7} and 1 in different runs. The results presented in Table 1 are for an typical case with $a = 2$, $b = 1$, $\varepsilon = 10^{-6}$. Reduction in residuals by nine orders of magnitude was achieved in each run.

Table 2 contains the number of iterations and runtimes in seconds for a three-dimensional model problem solved sequentially.

Table 3 contains the results of solving a typical model problem in parallel.

5 Conclusions

The capacitance matrix preconditioner for nonsymmetric matrices described in this paper has shown encouraging results in comparison with the established way of preconditioning. It possesses good scalability and convergence properties and can be applied to the problems with variable coefficients as well as constant ones.

The new method of preconditioning was designed and tested on problems in two and three dimensions discretised with the five-point upwinding on a regular grid. It is specifically intended for use in parallel computation and it has proved to possess two important qualities — naturally decoupled structure and closeness to the interface operator, resulting in accelerated convergence in many important industrial problems.

REFERENCES

[ACP81] Appleyard J. R., Cheshire I. M., and Pollard R. K. (1981) Special techniques for fully-implicit simulators. In *Proc. European Symposium on Enhanced Oil Recovery Bournemouth, UK*, pages 395–408.

[Dry82] Dryja M. (1982) A capacitance matrix method for Dirichlet problem on polygon region. *Numer. Math.* 39: 51–64.

[GM83] Golub G. H. and Mayers D. (1983) The use of preconditioning over irregular regions. In *Proc. 6th Internl. Conf. Comput. Meth. Sci. Engng. Versailles, France.*

[Val89] Valiant L. G. (August 1989) A bridging model for parallel computation. *Communications of the ACM* 33(8): 103–111.

[vdV92] van der Vorst H. A. (March 1992) BiCGStab: A fast and smoothly converging variant of BiCG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comp.* 13(2): 631–644.

64

Hybrid Newton-Krylov/Domain Decomposition methods for Compressible Flows

Moulay D. Tidriri

1 Introduction

Newton-Krylov methods have been shown to be very efficient for the solution of compressible flows [CGKT94], [Tid95a]-[Tid96]. On the other hand, domain decomposition methods provide efficient algorithms suitable for the parallel computing environment. In this study we are interested to two important classes of domain decomposition methods. The first class corresponds to the classical Schwarz-based domain decomposition methods. These methods reduce the solution of a given global problem into the solution of local problems and have potential applications on parallel computing environment. The second class is more recent and corresponds to the domain decomposition time marching algorithm [TT94]-[JFB96] and [Tid92]-[Tid95b]. This method was introduced initially to solve complex physical problems in which many different phenomenon occur. In this report we study the combination of these two classes of domain decomposition methods with Newton-Krylov matrix-free algorithms.

In the next section we study the first hybrid method. The study of the second hybrid method is performed in section 3. The last section is devoted to some conclusions.

2 First Hybrid Newton-Krylov/Domain Decomposition Methods

Euler Solver

The bidimensional Euler Equations in conservative form are written

$$W_t + F(W)_x + G(W)_y = 0,$$

where $W = (\rho, \rho u, \rho v, e)^T$, $F = (\rho u, \rho u^2 + p, \rho uv, u(e+p))^T$, and $G = (\rho u, \rho uv, \rho v^2 + p, v(e+p))^T$. In these expressions ρ is the density, u and v are the velocity components,

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

e is the internal energy, p is the pressure defined by $p = (\gamma - 1)(e - \rho(u^2 + v^2)/2)$, and finally, γ is a constant with $\gamma \approx 1.4$ for air. After transforming the variables into the curvilinear coordinates

$$\tau = t, \quad \xi = \xi(x, y), \quad \eta = \eta(x, y),$$

we obtain the following set of equations

$$\tilde{W}_\tau + (\tilde{F})_\xi + (\tilde{G})_\eta = 0, \tag{2.1}$$

where $\tilde{W}$ and the contravariant flux vectors, $\tilde{F}$ and $\tilde{G}$, are defined in terms of the Cartesian fluxes and the Jacobian determinant of the coordinate system transformation, through $\tilde{W} = J^{-1}W$, $\tilde{F} = J^{-1}(\xi_t W + \xi_x F + \xi_y G)$, $\tilde{G} = J^{-1}(\eta_t W + \eta_x F + \eta_y G)$, and $J = \frac{\partial(\xi,\eta,\tau)}{\partial(x,y,t)}$. An implicit finite volume discretization of the equation (2.1) together with a flux splitting approach yields the following nonlinear system

$$f(W^{n+1}) = 0. \tag{2.2}$$

A linearization of first order in time yields the standard defect-correction method

$$\mathbf{A}\delta W^n = b. \tag{2.3}$$

The different fluxes involved above are computed using Roe's approximate Riemann solver [Roe81]. In (2.3), the Jacobians are evaluated using Van Leer's scheme. In the fully implicit form the boundary conditions are implemented through: $\frac{\partial f_b}{\partial W}\delta W = -f_b(W)$. In this case, the CFL number may be adaptively advanced according to: $\mathrm{CFL}^{n+1} = \mathrm{CFL}^n \cdot \frac{||f(W)||^{n-1}}{||f(W)||^n}$, where the superscript refers to the iteration in time. For more details we refer to [MBW88] and [Tid95a].

Description of the Preconditioned Newton-Krylov matrix-free algorithms

The preconditioned Newton-Krylov matrix-free method [BS90], applied to the fully implicit nonlinear system (2.2), yields the following algorithm

- Define δW_0^n, an initial guess.
- For $k = 0, 1, 2, \cdots$ until convergence do

$$\text{Solve} \quad M^{-1}\frac{f(W_k^n + \epsilon\delta W_k^n) - f(W_k^n)}{\epsilon} = -M^{-1}f(W_k^n). \tag{2.4}$$

$$\text{Set} \quad W_{k+1}^n = W_k^n + \delta W_k^n.$$

The selection of the parameter ϵ is discussed in [Tid95b]. The preconditioner M^{-1} is constructed using an approximation similar to that used to derive the matrix $\mathbf{A}$ of the defect-correction procedure (2.3). This results in a combined discretization in which for each linear step (2.4) of the Newton iteration the preconditioner is not derived from the actual higher-order system. Instead, this preconditioner is derived using an approximation of the Jacobian matrix that employs a lower-order discretization in a similar fashion to defect-correction procedure. In this study the preconditioner corresponds to the parallel Schwarz domain decomposition preconditioner which will be described in the next section and the Krylov methods correspond to GMRES [SS86].

Additive and Multiplicative Schwarz methods

Considering an overlapping decomposition of the physical polygonal domain, the multiplicative Schwarz algorithm for the solution of the linear system (2.3) or (2.4) corresponds to:

$$(I - O_I)v = g, \tag{2.5}$$

with an appropriate g. Above, $O_I = (I - P_{N_{sd}}) \cdots (I - P_1)$, N_{sd} is the number of subdomains, and $P_i = R_i^T A_i^{-1} R_i A$. A_i are the local matrices and R_i are the algebraic restrictions while R_i^T are the algebraic extensions. The additive Schwarz method corresponds to

$$\sum_{i=1}^{N_{sd}} P_i v = g, \tag{2.6}$$

with an appropriate g.

Numerical Results

The test problem on which we study the performance of the methodology described above corresponds to a NACA0012 steady transonic airfoil at an angle of attack of 1.25 degrees and a freestream Mach number of 0.8 using the C-grids 128 × 32 cells. All calculations in this section are performed on the same Sparc10 machine. Since we are dealing with different methods which require varying amounts of work at each time step we believe that the CPU time is the only true measure for comparing them. The steady state regime is declared when the nonlinear residual norm reaches a value of (or less than) 10^{-5}. The Schwarz-based domain decomposition solver uses the PETSc library that was developed at Argonne National Laboratory [GS93]. In Table 1, we present the iteration count (number of nonlinear iterations) and CPU time (in seconds) for steady transonic flow at convergence using Schwarz algorithms in combination with defect correction procedures. The treatment of the boundary conditions is implicit and the CFL number is equal to 100. In Table 2, we present the iteration count and CPU time (in seconds) for steady transonic flow at convergence using Schwarz algorithms in combination with Newton-Krylov matrix-free methods. The treatment of the boundary conditions is also implicit and the starting CFL number is 30. Comparing the two tables, we observe that the additive Schwarz algorithm combined with Newton-Krylov matrix-free method reduces the CPU time by almost 50% for the various decompositions studied here, as compared to its combination with defect correction procedures (see [Tid96]).

Table 1 Schwarz methods combined with defect-correction procedures.

	Block Jacobi		Add. Schwarz		Mult. Schwarz	
Decomp.	Iter	CPU	Iter	CPU	Iter	CPU
2×2	547	8911	553	11096	566	9123
4×4	540	9114	552	11899	577	9717
8×8	539	11430	546	16215	574	11482

Table 2 Schwarz methods combined with preconditioned Newton-Krylov matrix-free methods.

	Block Jacobi		Add. Schwarz		Mult. Schwarz	
Decomp.	Iter	CPU	Iter	CPU	Iter	CPU
2×2	31	5474	31	6102	33	8409
4×4	32	5384	28	5708	30	4759
8×8	32	6594	35	7493	25	4106

3 Second Hybrid Newton-Krylov/Domain Decomposition Methods

Navier-Stokes Equations

Let us consider the compressible Navier-Stokes equations which we formally write either as

$$\frac{\partial W}{\partial t} + div[F(W)] = 0 \text{ on } \Omega \text{ (conservative form)}$$

or as

$$\frac{\partial U}{\partial t} + T(U) + D(U) = 0 \text{ on}\Omega \text{ (non conservative form)}$$

with $W = (\rho, \rho v, \rho E)$ and $U = (\rho, v, \theta)$ as the conservative and nonconservative variables, $F = F_C + F_D$ as the total flux (convective and viscous part), and T and D the convective and viscous terms in the nonconservative form of the Navier-Stokes equations. The problem consists in computing a steady solution of these equations, with boundary conditions

$$\rho v, \rho E \text{ given on } \Gamma_e \text{ (exterior limit of the domain),}$$

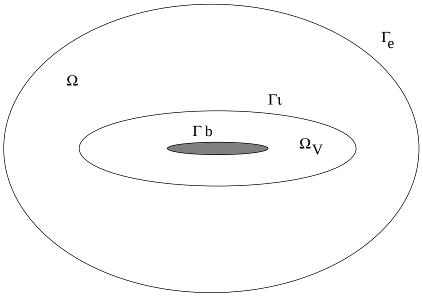
$$\rho \text{ given on } \Gamma_e \cap \{x, v(x) \cdot n \leq 0\} \text{ (inflow),}$$

$$v = 0 \text{ on the body } \Gamma_b, \text{ (no slip),}$$

$$\theta = \theta_b \text{ on the body } \Gamma_b.$$

The strategy discussed below couples a *global conservative scheme*, defined on the whole domain, and based on a finite volume space discretization [RS88], and *a local*

Figure 1 The global geometry



approximation, defined in the neighborhood of the body, which is presently based on a mixed Finite Element approximation of the nonconservative Navier-Stokes equations [BGD$^+$89].

The General Coupling Strategy

For coupling external Navier-Stokes equations, with local Navier-Stokes equations, we introduce two domains, a global one Ω, a local one Ω_V included in Ω, and an interface Γ_i (Fig. 1). The global solution W on Ω and the local solution U_{loc} on Ω_V, which both satisfy the Navier-Stokes equations, are matched by the following boundary conditions, inspired of Schwarz overlapping techniques :

$$\begin{cases} W = \text{ given imposed value on } \Gamma_e, \\ n \cdot \sigma(W) \cdot \tau = n \cdot \sigma(U_{loc}) \cdot \tau \text{ on the body } \Gamma_b, \text{ (equality of friction forces)} \\ q(W) \cdot n + n \cdot \sigma(W) \cdot v = q(U_{loc}) \cdot n \text{ on } \Gamma_b, \text{ (equality of total heat fluxes)} \\ v \cdot n = 0 \text{ on } \Gamma_b, \\ U_{loc} = 0 \text{ on } \Gamma_b \; U_{loc} = W \text{ on the interface } \Gamma_i. \end{cases}$$

Above, $n.\sigma.n$ and $n.\sigma.\tau$ respectively denote the normal and the tangential force exerted by the body on the flow, with n the unit normal vector to the body oriented towards its interior.

The calculation of U_{loc} and W satisfying the above boundary conditions is then obtained by the time marching algorithm, which was introduced in [TT94]-[JFB96] and [Tid92]-[Tid95b]) and which leads to the following algorithm : **Initialization**

1. Guess an initial distribution of the conservative variable W in the global domain

Ω ;

2. Advance in time this distribution by using the global Navier-Stokes solver on N_1 time steps, with *Dirichlet* type boundary conditions on the body Γ_b ;

3. Deduce from this result an initial distribution of the local variable U_{loc} on the interface Γ_i and in the local domain Ω_V ;

4. Advance in time this distribution by using the local solver on N_2 time steps with Dirichlet boundary conditions on Γ_i and Γ_b.

Iterations

5. From U_{loc}, compute the friction forces $n \cdot \sigma(U_{loc}) \cdot \tau$ and heat flux $q(U_{loc}) \cdot n$ on the body Γ_b ;

6. Advance the global solution in time (N_1 steps) by using the global Navier-Stokes solver with the above viscous forces as boundary conditions on Γ_b;

7. From W, compute the value of U_{loc} on the interface Γ_i ;

8. Using this new value as Dirichlet boundary conditions on Γ_i, advance the local solution in time (N_2 steps) and go back to step 5 until convergence is reached.

This algorithm completely uncouples the local and the global problems, which can therefore be solved by independent solvers. A parallel version is also quite possible although it is generally wiser to use parallel solvers within steps 6 and 8.

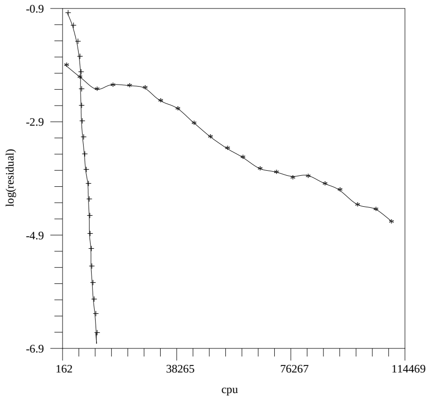
Global and Local Solvers

To solve the global conservative Navier-Stokes equations we use the hybrid finite volume/finite element method in which the convective flux is computed by an Osher approximate Riemann solver. The resulting linear system is solved by a block relaxation method. The local nonconservative Navier-Stokes equations are discretized by mixed finite elements (P_1 for ρ and θ, P_1 on the subdivided P_2 grid for the velocity). The resulting nonlinear local system is solved by using the preconditioned Newton-Krylov matrix-free method described in 2 with diagonal preconditioner. We refer to [Tid95a] for more details.

Numerical Results

The test problem consists of a two dimensional flow around an ellipse, with 0 angle of attack, $M_\infty = 0.85$, Reynolds number $= 100$, and a wall temperature $T_W = 2.82 T_\infty$. First, we have calculated the Navier-Stokes solution employing the global nonconservative solver alone on a mesh that has 4033 nodes and 7942 elements for the P_1 grid and 16184 nodes and 32120 elements for the grid P_2. We then performed a calculation using the coupling algorithm described above on a global mesh that has 1378 nodes and 2662 elements and a local mesh that has 1114 nodes and 4282 elements. On Figure (2) we show a CPU time comparisons of the two calculations which were performed on an apollo DN 10000. This figure shows the excellent performance of Newton-Krylov matrix-free method used to solve the local model through the domain decomposition time marching algorithm as compared to its use in the standard approach (see [Tid95a]).

Figure 2 CPU time comparisons between the uncoupled scheme and the coupled approach. Above the + curv corresponds to the coupled scheme and the * one corresponds to the uncoupled nonconservative scheme.



4 Conclusions

In this study we have studied the combination of Newton-Krylov matrix-free algorithms with two classes of domain decomposition methods. In both cases we have given numerical applications to compressible Euler and Navier-Stokes equations that illustrate the performance of the resulting hybrid algorithms.

Acknowledgement

This work was supported by the National Aeronautics and Space Administration under NASA contract NAS1-19480 while the author was in residence at the Institute for Computer Applications in Science and Engineering.

REFERENCES

[BGD+89] Bristeau M., Glowinski R., Dutto L., G. J. P., and Rogé (1989) Compressible viscous flow calculations using compatible finite element approximations. In *7th Int. Conf. on Finite Element Methods in Flow Problems.* Huntsville, Alabama.

[BS90] Brown P. N. and Saad Y. (1990) Hybrid krylov methods for nonlinear systems of equations. *SIAM J. Sci. Stat. Comp.* 11: 450–481.

[CGKT94] Cai X.-C., Gropp W. D., Keyes D. E., and Tidriri M. D. (1994) Newton-krylov-schwarz methods in cfd. In Rannacher R. (ed) *Proceedings of the International Workshop on the Navier-Stokes Equations, Notes in Numerical Fluid Mechanics.* Braunschweig.

[GS93] Gropp W. D. and Smith B. F. (1993) Simplified linear equations solvers users manual. Technical Report ANL 93/8, Argonne National Laboratory.

[JFB96] J. F. Bourgat P. Le Tallec M. D. T. (1996) Coupling navier-stokes and boltzmann. *J. Comp. Phy.* 127: 227–245.

[MBW88] Mounts J. S., Belk D. M., and Whitfield D. L. (September 1988) Program eagle user's manual, vol. iv – multiblock implicit, steady-state euler code. Technical Report TR-88-117, Air Force Armament Laboratory", TR-88-117, Vol. IV, September 1988.

[Roe81] Roe P. L. (1981) Approximate riemann solvers, parameter vector, and difference schemes. *J. Comp. Phy.* 43: 357–372.

[RS88] Rostand P. and Stoufflet B. (July 1988) Finite volume galerkin methods for viscous gas dynamics. Technical Report RR-863, INRIA.

[SS86] Saad Y. and Schultz M. H. (1986) Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear. *SIAM J. Sci. Stat. Comp.* 7: 865–869.

[Tid92] Tidriri M. D. (1992) *Couplage d'approximations et de modèles de types différents dans le calcul d'ecoulements externes.* PhD thesis, Université de Paris-Dauphine.

[Tid95a] Tidriri M. D. (1995) Domain decompositions for compressible navier-stokes equations. *J. Comp. Phy.* 119: 271–282.

[Tid95b] Tidriri M. D. (June 1995) Krylov methods for compressible flows. Technical Report CR 95-48, ICASE.

[Tid96] Tidriri M. D. (January 1996) Schwarz-based algorithms for compressible flows. Technical Report CR 96-4, ICASE.

[TT94] Tallec P. L. and Tidriri M. D. (October 1994) Convergence of domain decomposition algorithms with full overlapping for the advection-diffusion problems. Technical Report RR-2435, INRIA.

65

Multidomain Finite Elements and Finite Volumes for Advection-Diffusion Equations

M.-C. Ciccoli, R. L. Trotta

1 Introduction

New interface conditions are proposed for domain decomposition methods in the advectively dominated limit of advection-diffusion problems.

Domain decomposition methods are interesting for several reasons. They allow a simplification of the geometry, a reduction of the size of the problems, and the use of different physical and/or numerical models on the different subdomains in order to get a more accurate modelization of the flow. Furthermore DD methods are easily parallelizable.

Unfortunately, DD algorithms that work well for viscosity dominated flows can perform very badly for convectively dominated flows. This is due to the fact that the matching conditions, although mathematically correct, may not respect the hyperbolic limit of the advection-diffusion equation.

The adaptive DD algorithms enforce the appropriate interface condition in respect with the "direction of the wind". We are interested in the capability of these algorithms to efficiently solve an advection-diffusion equation in view to apply them to the Navier-Stokes equations.

Experiments in this article have been made using two different discretization methods: finite element and finite volume/finite element.

We investigate the properties of the Adaptive Dirichlet Neumann (ADN) and the Adaptive Robin Neumann (ARN) algorithms, proposed by Carlenzoli and Quarteroni [CQ95]. We also study the performances of two new algorithms denoted d-ADN and d-ARN which are *damped* versions of the ADN and ARN algorithms and have been constructed to improve the convergence of the ADN and ARN algorithms. We will see that the d-ADN and d-ARN algorithms do not get exactly the right solution, but the error they introduce is acceptable for small diffusion coefficient which is the case we are interested in and they converge significantly faster than the ADN and ARN algorithms.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Regarding other recent works on domain decomposition methods for advection-diffusion problems, we mention [NR95], and [TB95]. These two works are made in the framework of finite difference discretizations. In [NR95] A Schur type formulation with outflow boundary conditions using overlapping subdomains decomposition is constructed, while in [TB95] a way to improve the Schwarz algorithm convergence for advection-dominated cases is proposed.

2 The Advection-Diffusion Boundary Value Problem

Let Ω be a bounded, connected, open subset of $I\!R^2$ with a Lipschitz continuous boundary $\partial\Omega$ and denote by $\mathbf{n}$ the unit outward normal direction on $\partial\Omega$. Let $\mathbf{b} = \mathbf{b}(\mathbf{x})$ denote the given flow velocity, $\varepsilon = \text{const} > 0$ denote the diffusivity, and $a \geq 0$ the absorption coefficient. Let $\{\partial\Omega^{in}, \partial\Omega^{out}\}$ be a partition of $\partial\Omega$, where

$$\partial\Omega^{in} = \{\mathbf{x} \in \partial\Omega \;\mid\; \mathbf{b}\cdot\mathbf{n} < 0\} \qquad \textit{inflow boundary}$$

$$\partial\Omega^{out} = \partial\Omega \setminus \partial\Omega^{in} \qquad \textit{outflow boundary}.$$

The *scalar steady advection-diffusion boundary value problem* consists of finding $u = u(\mathbf{x})\ \forall\mathbf{x} \in \bar{\Omega}$ such that

$$(AD) \left\{ \begin{array}{ll} L_\varepsilon u := -\varepsilon\Delta u + div(\mathbf{b}u) + au = f & \text{in } \Omega \\ & \\ u = g & \text{on } \partial\Omega\,, \end{array} \right. \tag{2.1}$$

where the given body source $f : \Omega \to \mathbf{R}$ and $g : \partial\Omega \to \mathbf{R}$ are prescribed data.

In the advection-dominated regime, i.e. when $\dfrac{\varepsilon}{\|\mathbf{b}\|} \ll 1$ the solution u will vary rapidly in layers of width $O(\varepsilon)$ at the outflow boundary $\partial\Omega^{out}$ and in layers of width $O(\sqrt{\varepsilon})$ across characteristic curves issuing from discontinuous boundary data.

If layers arise, the numerical method may produce oscillatory solution, if this situation is not properly faced, resorting either to a stabilization method, like SUPG, GALS, DW (see [BBF+92]) or to an upwind formulation.

DD methods are interesting for such problems because, beyond the classical reasons, the problem can be stabilized only in the subregions where it is necessary.

3 Adaptive DD Methods

To simplify our presentation, we consider a partition of the initial domain Ω into two nonoverlapping subdomains, Ω_1 and Ω_2. The generalization to several subdomains is straightforward. Let Γ be the interface between Ω_1 and Ω_2, $\Gamma = \partial\Omega_1 \cap \partial\Omega_2$. Let u_i be the restriction of u on Ω_i and $\mathbf{n}_i$ the outward normal unit vector on Γ.

The multidomain formulation can be obtained by decoupling the resolution in Ω_1 from the resolution in Ω_2, and by assigning properly matching conditions on the interface Γ. Natural transmission conditions arise from the requirement that u is the solution to the overall problem (hence, in particular, u is sought in a precise functional space), and that u_k must be the restriction of u to Ω_k, $k = 1, 2$.

For elliptic problems these conditions are the continuity of the solution and the continuity of the flux across the interface Γ.

The well known Dirichlet/Neumann algorithm (see [BW86] and [MQ89]) assigns, at each iteration of the procedure, a Dirichlet condition to one subdomain and a Neumann condition to the other one.

It works well for viscous flows, but for advection dominated problems can produce at each iteration nonphysical layers because the matching condition may not respect the hyperbolic limit of the advection-diffusion equation. The Dirichlet condition at the outflow, prescribing specific values, can generate artificial layers.

This idea is the basis of the adaptive DD methods proposed by Carlenzoli and Quarteroni in 1993. The first method proposed is the ADN method; in this algorithm, a Dirichlet condition is imposed at the part of the interface (on Γ_i^{in}) where the "flow is coming in" the domain, inversely a Neumann condition is used where (on Γ_i^{out}) the "flow is going out of" the domain. The Adaptive Dirichlet/Neumann algorithm starts with (u_1^0, u_2^0) and constructs the iterates (u_1^k, u_2^k) by solving the following problems:

$$\text{(ADN)} \qquad \begin{cases} L_\varepsilon u_1^k = f & \text{in } \Omega_1 \\ u_1^k = \theta u_2^{k-1} + (1-\theta) u_1^{k-1} & \text{on } \Gamma_1^{in} \\ \dfrac{\partial u_1^k}{\partial n_1} = -\dfrac{\partial u_2^{k-1}}{\partial n_2} & \text{on } \Gamma_1^{out} \end{cases} \qquad \begin{cases} L_\varepsilon u_2^k = f & \text{in } \Omega_2 \\ u_2^k = \theta u_1^{k} + (1-\theta) u_2^{k-1} & \text{on } \Gamma_2^{in} \\ \dfrac{\partial u_2^k}{\partial n_2} = -\dfrac{\partial u_1^{k}}{\partial n_1} & \text{on } \Gamma_2^{out} \end{cases} \qquad (3.2)$$

θ is a relaxation parameter introduced to improve the convergence of the algorithm.

The other adaptive domain decomposition method, ARN, arises from the fact that for advection dominated problems the continuity has to be weakly enforced [Qua90]. Thus, in the ARN algorithm, the Dirichlet condition is replaced by a Robin condition, that means we impose:

$$\varepsilon \frac{\partial u_1}{\partial \mathbf{n}_1} - \mathbf{b} \cdot \mathbf{n}_1 u_1 = -\varepsilon \frac{\partial u_2}{\partial \mathbf{n}_2} + \mathbf{b} \cdot \mathbf{n}_2 u_2 \qquad \text{instead of} \qquad u_1 = u_2 \qquad \text{on } \Gamma_i^{in}$$

The two approaches, ADN and ARN, are equivalent if $\varepsilon \to 0$ and $\Gamma_i^0 = \{\mathbf{x} \in \Gamma \mid \mathbf{b} \cdot \mathbf{n}_i = 0\}$ is negligible. For more details on the two algorithms, see [Tro96], [Cic].

The Damped Versions

It is well known that for pure hyperbolic equations (i.e. $\varepsilon = 0$) the boundary conditions have to be given only on the inflow part of the boundary. As far as domain decomposition methods are concerned, one has to give also the transmission conditions, but only on the inflow part of the interface Γ. But, even in the case $\varepsilon = 0$, the DD algorithms enforce a derivative continuity at the outflow part of the interface. This slows down the convergence rate of the algorithms.

This remark leads to the construction of two new algorithms denoted d-ADN (damped ADN) and d-ARN (damped ARN) by changing the Neumann condition of the normal derivative continuity into the hyperbolic outflow condition.

Of course, the damped algorithms do not provide exactly the right solution for $\varepsilon \neq 0$. In particular, the error between two d-ARN solutions at the interface satisfies the inequality

$$\int_\Gamma \left(b.n_1 - \frac{\varepsilon}{2}\right)(u_1 - u_2)^2 \leq \frac{\varepsilon}{2}\int_\Gamma \left(\frac{\partial u_2}{\partial n_2}\right)^2$$

This error decreases proportionally to ε and is, obviously, equal to zero for $\varepsilon = 0$.

We are satisfied if the order of the error introduced by the damped formulation is the same than this of the discretization error.

We refer to [Tro96] and [Cic] for a more complete analysis of the errors of the damped methods.

The Discretized DD Algorithms

We do not give extended explanations for the discretizations of the DD algorithms. See [Tro96] for the finite element discretization and to [Cic] for the finite volume/finite element discretization. In the finite element approximation we discretize the advection-diffusion equation using a P1 Galerkin method. In the mixed finite volume/finite element formulation, we use an upwind finite-volume approximation for the convective term and a finite-element approximation using the P_1 basis function for the diffusive term.

4 Validation on Test Cases

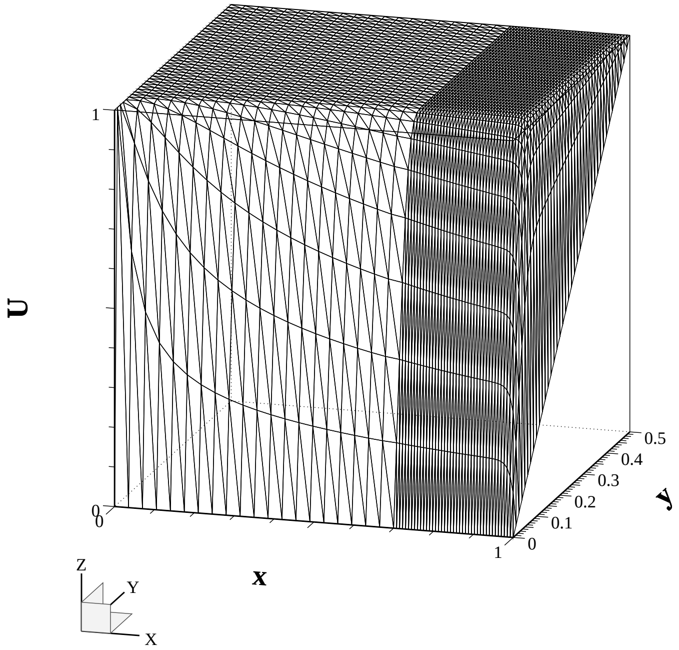
The schemes are implemented on a cluster of an IBM RS/6000 (model 560 and 950) workstations connected by Ethernet, as well as on IBM-9706-SP1 parallel distributed memory machine. We have used both the finite element and finite volume/finite element discretizations.

Extended experiments have been made to test the capability of the adaptive algorithms to solve convection-dominated problems. We refer to [Tro96] and [Cic] for details. We only give here the conclusions we draw from the test cases we computed. We were particularly interested in the dependence of the algorithms on the diffusion parameter ε, on the mesh size, on the position of the interfaces, on the number of subdomains and on the presence of crosspoints in the decomposition.

We notice that the convergence of the ADN and ARN algorithms does not depend on ε, when sufficiently small (which are the cases we are interested in). The convergence is also not sensitive to mesh size. On the contrary, the convergence depends on the position of the interface and on the number of subdomains. Such is the general behaviour for all the algorithms.

The choice of the relaxation parameter is very important. In fact a good choice of θ can accelerate the convergence rate. An analysis of the best value of θ is made in [GGQ]. We observe that, for ADN the number of iterations needed to reach convergence is always lower than for ARN, but for ADN θ changes a lot with different problems, ε, and number of subdomains. On the contrary, for ARN, in the advection dominated cases, the best convergence is always achieved for $\theta = 1$, making this method easier to use.

Figure 1 Numerical solution of the thermal boundary layer problem



For the damped algorithms (d-ADN and d-ARN) the convergence depends on ε, even when small. But the number of iterations to reach convergence is quite small, sometimes twenty times lower than this needed with the ADN or ARN algorithms. The d-ARN algorithm has a bit faster convergence than the d-ADN.

On the two following paragraphs we present two test cases to illustrate the behaviour of the algorithms. The first computation uses a finite element discretization, the second a finite volume/finite element discretization.

Thermal Boundary Layer Problem

This classical benchmark problem, undertakes to find the solution of

$$-\varepsilon \Delta u + 2yu_x = 0 \quad \text{on} \quad \Omega = (0,1) \times (0,0.5)$$

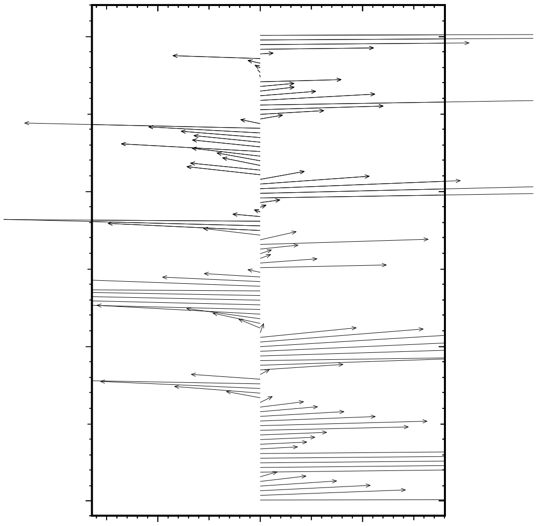
with boundary conditions $u = 1$ for boundary sides $x = 0$ and $y = 0.5$, $u = 0$ for boundary side $y = 0$ and $u = 2y$ for $x = 1$. The problem has two zones of large gradient.

We divide the domain in two subdomains:

$\Omega_1 = (0,0.7) \times (0,0.5)$ with 21×41 uniformly spaced grid points

$\Omega_2 = (0.7,1) \times (0,0.5)$ with 41×41 uniformly spaced grid points.

Figure 2 Velocity field at an interface



We apply ADN, ARN and d-ARN to problem with $\varepsilon = 10^{-4}$, and we obtain the numerical solution plotted in Fig. 1. The number of iteration needed is 14 for ADN ($\theta = 0.83$), 19 for ARN ($\theta = 1$) and 2 for d-ARN ($\theta = 1$).

Unsteady Calculation — Reservoir Problem

In this section, we show how to extend the algorithms to unsteady calculations. We solve the unsteady advection-diffusion equation:

$$\frac{\partial u}{\partial t} - \varepsilon \Delta u + \mathrm{div}(bu) = 0$$

The first-order Euler implicit scheme applied to the previous problem is:

$$\frac{u^{n+1} - u^n}{\Delta t} - \varepsilon \Delta u^{n+1} + \mathrm{div}(bu^{n+1}) = 0$$

or:

$$\frac{u^{n+1}}{\Delta t} - \varepsilon \Delta u^{n+1} + \mathrm{div}(bu^{n+1}) = \frac{u^n}{\Delta t}$$

We identify the usual formulation by denoting: $a = \dfrac{1}{\Delta t}$, $\quad f = \dfrac{u^n}{\Delta t}$. We apply the DD algorithms at each time iteration.

The test case we present now models the saturation of oil in a porous media. A Dirichlet condition ($u = 1$) is imposed on the boundary $y = 0$. On the three other sides, homogeneous Neumann conditions are imposed. The initial conditions are:

$$\begin{cases} u = 1 & \text{on } y = 0 \\ u = 0 & \text{elsewhere.} \end{cases}$$

For this calculation, the viscosity ε is taken equal to 10^{-4}. The velocity field at one of the interfaces can be seen on Fig. 2. We observe that the direction of the velocity changes very often at the interfaces. This is a rather difficult test for the Adaptive DD algorithms.

The entire mesh has 101 × 101 points. We use a partition in five subdomains. The interfaces are situated at $x = 0.2$, $x = 0.4$, $x = 0.6$, $x = 0.8$, and are parallel to the flow direction. Seven hundred and ninety two points are situated on the interfaces.

At each time step, the d-ADN algorithm requires seven or eight iterations to reach the 10^{-4} convergence stopping criterion. We refer to [Cic] to have more details on the test case solution. By making this calculation, we wanted to know the behaviour of the Adaptive DD algorithms in a real test case in which the interface treatment is particularly complex. And we observed that the d-ADN algorithm is quite efficient and needs a low and constant number of iterations along the time steps.

5 Conclusion

For advection dominated advection-diffusion problems, adaptive methods are certainly superior, because they take into account the direction of the "wind".

By looking at the results obtained, we can conclude that the ADN method is not as robust as the ARN, because of the difficulties arising in the choice of the best parameter θ. But, a theoretical convergence analysis of the ADN algorithm would provide a way to choose the optimal θ and ADN could be more efficient than ARN, in terms of number fo iterations, when ε becomes small. It must be also underlined that ADN performs well even for large ε. The damped versions of the ADN and ARN algorithms accelerate convergence and reduce very significantly the number of iterations needed for a computation. For problems with sufficiently small diffusion, the d-ADN algorithm will provide a faster convergence. And if a discontinuous solution is acceptable, the d-ARN algorithm seems to be the most suited method.

In the first step, we have tested the performances of these adaptive algorithms on an advection-diffusion model problem to analyze their behaviour.

Several developments of our work are possible. One could be the adaptation of the algorithms to the Navier-Stokes equations and to the coupling of the Euler and Navier-Stokes equations. Another could be the use of adaptive methods as preconditioners. And, of course, the development of a mathematical theory would be very helpful.

REFERENCES

[BBF$^+$92] Brezzi F., Bristeau M.-O., Franca L. P., Mallet M., and Rogé G. (1992) A relationship between stabilized finite element methods and the Galerkin method with bubble functions. *Comp. Meths. Appl. Mech. Eng.* 96: 117–129.

[BW86] Bjørstad P. E. and Widlund O. B. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal.* 23: 1097–1120.

[Cic] Ciccoli M.-C.Adaptive domain decomposition algorithms and finite volume/finite element approximation for advection-diffusion equations. Submitted to Journal of Scientific Computing.

[CQ95] Carlenzoli C. and Quarteroni A. (1995) Adaptive domain decomposition methods for advection - diffusion problems. In Babuska I. e. a. (ed) *Modeling, Mesh Generation, and Adaptive Numerical Methods for Partial Differential Equations*, volume 75 of *IMA Volumes in Mathematics and its Applications*, pages 165–199. Springer Verlag edition.

[GGQ] Gastaldi F., Gastaldi L., and Quarteroni A.Adaptive domain decomposition methods for advection dominated equations. To appear on East-West J. Numer. Math.

[MQ89] Marini L. D. and Quarteroni A. (1989) A relaxation procedure for domain decomposition methods using finite elements. *Numer. Math.* 55: 575–598.

[NR95] Nataf F. and Rogier F. (1995) Outflow boundary conditions and domain decomposition method. In Keyes D. E. and Xu J. (eds) *Proc. Seventh Int. Conf. on Domain Decomposition Meths.*, volume 180 of *Contemporary Mathematics*, pages 289–293. American Mathematical Society edition.

[Qua90] Quarteroni A. (1990) Domain decomposition methods for system of conservation laws: spectral collocation. *SIAM J. Sci. Stat. Comput.* 11: 1029–1052.

[TB95] Tan K. H. and Borsboom M. J. (1995) On generalized Schwarz coupling applied to advection-diffusion problems. In Keyes D. E. and Xu J. (eds) *Proc. Seventh Int. Conf. on Domain Decomposition Meths.*, volume 180 of *Contemporary Mathematics*, pages 289–293. American Mathematical Society edition.

[Tro96] Trotta R. L. (1996) Multidomain finite elements for advection-diffusion equations. *Appl. Numer. Math.* 21: 91–118.

66

A Funaro-Quarteroni Procedure for Singularly Perturbed Elliptic Boundary Value Problems

M. Garbey, L. Viry, O. Coulaud

1 Introduction

We analyze the Funaro-Quarteroni alternative procedure for the solution of singular perturbation problems. We show that for an appropriate choice of the domain decomposition, one obtains a fast convergent iterative scheme with *no relaxation* that resolves the boundary layers. The convergence is superlinear with respect to the singular perturbation parameter in the following sense: the amplification factor is $o(\epsilon)$. We give sharp estimates of the interface position and convergence rates for homogeneous domain decomposition in one dimensional space. This analyse can be generalized in a two dimensional space on a disk ([GVC96]). We extend our results to heterogeneous domain decomposition arising in a simplified model of an electromagnetic problem. Our method has been implemented with finite difference approximations and finite element codes (Modulef).

2 Boundary Layers in One-dimensional Space

Homogeneous Domain Decomposition

In this section, we consider a linear second-order singular perturbation problem of the following type:

$$\left\{ \begin{array}{l} L_\epsilon \phi = -\epsilon \phi" + \phi = F \ in \ \Omega = (0,1); \\ \phi(0) = \alpha_0 \ ; \ \phi(1) = \alpha_1. \end{array} \right. \tag{2.1}$$

ϵ is a small positive parameter, $\epsilon \in]0, \epsilon_0]$ for some $\epsilon_0 > 0$. Problems of this type exhibit boundary layers usually at both ends of the interval. This trivial one dimensional problem will be used as a motivation for our method. In order to get fast convergence for the Funaro-Quarteroni iterative solver(F.Q) with *no relaxation* parameter, the domain decomposition must be properly designed. We restrict ourselves to the case of a single boundary layer in the neighborhood of 1. According to the

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

asymptotic analysis we should split the domain Ω into two subdomains $\Omega_{inner} = (a, 1)$ and $\Omega_{outer} = (0, a)$ where $a > 0$. Ω_{inner} covers the boundary layer at 1 and Ω_{outer} covers the domain of validity of the regular approximation. In order to make it easier to get sharp estimates in the maximum norm, we are going to use the finite difference framework. We keep the mesh in each subdomain regular and adapt the domain decomposition to the boundary layer stiffness. This should be very efficient on a parallel computer. Let us denote h_1 (respectively h_2) the mesh size on Ω_{outer} (respectively Ω_{inner}). Let us denote $L^{h_i},\ i = 1, 2$ the discretized operator that corresponds to L_ϵ. We will also restrict ourselves to the case where we have the same asymptotic order of grid points in each subdomain, i.e

$$h_1 \approx \frac{h_2}{1-a} \approx \frac{1}{N},$$

in order to balance the amount of work in each subdomain.

Dirichlet-Neumann Scheme

To solve (2.1), we introduce the following iterative procedure [FQZ88]

$$\left\{ \begin{array}{l} L^{h_1} \phi^p_{outer} = F \ in \ \Omega_{outer}; \\ \phi^p_{outer}(0) = \alpha_0 \ ; \ \phi^p_{outer}(a) = \phi^p_{inner}(a) \\ L^{h_2} \phi^{p+1}_{inner} = F \ in \ \Omega_{inner}; \\ \phi^{p+1}_{inner}(0) = \alpha_1 \ ; \\ \dfrac{\phi^{p+1}_{inner}(a+h_2) - \phi^{p+1}_{inner}(a)}{h_2} = \dfrac{\phi^p_{outer}(a) - \phi^p_{outer}(a-h_1)}{h_1} \end{array} \right. \tag{2.2}$$

To start the scheme, we impose an artificial boundary condition at point a. We use the same finite difference scheme in each subdomain with Dirichlet boundary condition at a in Ω_{outer} and Neumann boundary condition at a in Ω_{inner}.

We will proceed with the analysis of this iterative method in three steps: firstly we define the best interface location between the subdomains, based on a truncation error analysis, secondly we derive from the stability property of the discretized operator the rate of damping of the artificial boundary condition error. Lastly we combine these two results to get an estimate of convergence of the iterative solver to the *exact* solution of the differential problem (2.1).

The technique of demonstration is quite elementary but uses two types of small parameters: first the space steps h_1, second the small singular perturbation parameter ϵ. Our goal is to find the best path in the parameter space (h_1, ϵ) which provides superlinear convergence and optimal uniform approximation.

- **First Step: interface position**

We wish to determine the optimal interface position a, which minimizes the maximum error in both subdomains under the constraint that we have the same asymptotic order of mesh points N inside each subdomain. In this part, we neglect the artificial boundary condition error inherent to the Funaro-Quarteroni alternate $(F.Q)$ procedure. This error will be taken care of later on.

Let ϕ_{outer}(respectively ϕ_{inner}) be the restriction of ϕ to Ω_{outer} (respectively Ω_{inner}).

We define the following errors :

$$e_{outer} = \max_{\Omega_{outer}} |\phi_{outer} - \phi^{h_1}_{outer}|$$
$$e_{inner} = \max_{\Omega_{inner}} |\phi_{inner} - \phi^{h_2}_{inner}|$$

A classical center finite difference scheme applied to $-\epsilon u'' + u = f$ with exact Dirichlet boundary conditions gives

$$e_{outer} \approx \epsilon h_1^2 a^2 \max_{\Omega_{outer}} |\tfrac{d^{(4)}\phi}{dx^4}|$$

The analysis of the inner subdomain approximation with mixed exact boundary conditions gives two truncation errors, which we should consider in addition to the discretisation error of the Neumann boundary condition. We have

$$e_{inner} \approx \epsilon h_2^2 (1-a)^2 \max_{\Omega_{inner}} |\frac{d^{(4)}\phi}{dx^4}| + h_2^2 \frac{R}{1-R} \max_{\Omega_{inner}} |\frac{d^{(2)}\phi}{dx^2}|, \qquad (2.3)$$

where $R = 1 + \frac{h_2^2}{2\epsilon} + \frac{h_2}{\sqrt{\epsilon}}(1 + \frac{h_2^2}{2\epsilon})^{\frac{1}{2}}$.

We first notice that the truncation errors defined above depend strongly on the property of the solution that we want to approximate in each subdomain.

Let ϕ_0 be the outer expansion of ϕ and $\Theta(x,\epsilon)$ be the corrector i.e

$$\Theta(x,\epsilon) = \phi(x,\epsilon) - \phi_0(x,\epsilon) \approx exp(-\eta),$$

in the boundary layer with $\eta = \frac{1-x}{\epsilon}$. We show that the truncation error is dominated by the behavior of the corrector as in ([Gar96]).

Secondly we remark that the error in both subdomains is coupled because the Neumann boundary condition for the *inner* domain is only an approximation of a derivative in the *outer* domain. Thys we need to compute directly the error between the exact solution of the continuous problem and the formal limit of (2.2) when $p \to \infty$.

Lemma 1 *Let $\tilde{\phi} = (\tilde{\phi}_{i,j})_{i=0\ldots N, j=1,2}$ be the solution of the following linear system.*

$$\begin{cases} L^{h_1}\tilde{\phi}_{i,1} = F \quad i = 1, \ldots N-1, \\ L^{h_2}\tilde{\phi}_{i,2} = F \quad i = 1, \ldots N-1, \\ \tilde{\phi}_{0,1} = \alpha_0 \ ; \ \tilde{\phi}_{N,1} = \tilde{\phi}_{0,2}; \ \tilde{\phi}_{N,2} = \alpha_1, \\ \dfrac{\tilde{\phi}_{1,2} - \tilde{\phi}_{0,2}}{h_2} = \dfrac{\tilde{\phi}_{N,1} - \tilde{\phi}_{N-1,1}}{h_1}, \end{cases}$$

$$\textit{where} \qquad L^h \phi_i = -\epsilon \tfrac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{h} + \phi_i.$$

Let M be the composite grid $M = M_{outer} \cup M_{inner}$, with

$$\begin{cases} M_{outer} = \{x_{i,1} = i(\frac{a}{N}); i = 0 \ldots N\} \\ M_{inner} = \{x_{i,2} = a + i(\frac{1-a}{N}); i = 0 \ldots N\} \end{cases}$$

Let us suppose that: $N^{-1} \approx \sqrt{\epsilon}\delta$ with $\delta \succ\succ 1$ and $\| \ \|_\infty$ be the maximum norm on the composite grid M.

Under the previous hypothesis concerning the discretization and approximation of the operators in each subdomain, $\| \phi - \tilde{\phi}\|_\infty$ is asymptotically minimum when $1 - a \sim \sqrt{\epsilon}\log(\epsilon^{-2})$

• **second step: Damping of artificial boundary errors**

The convergence of the method depends essentially on the way an error which is introduced at the artificial interface propagates inside the subdomain.

In [Gar96] it is shown that we may have fast convergence with relatively small overlap even if we apply the straightforward Schwarz alternate procedure with Dirichlet boundary conditions. We will prove that the F.Q procedure may also have fast convergence and that the choice of boundary conditions is critical.

Let us consider the F.Q iterative procedure applied to the following homogeneous problem:

$$\left\{\begin{array}{c} L_1^h e_{i,1}^p = 0 \ ; \\ e_{0,1}^p = 0 \ ; \ e_{N,1}^p = e_{0,2}^{p-1} \ ; \\ L_2^h e_{N,2}^p = 0 \\ \frac{e_{1,2}^p - e_{0,2}^p}{h_2} = \frac{e_{N,1}^p - e_{N-1,1}^p}{h_1} \ ; e_{N,2}^p = 0 \ ; \end{array}\right.$$

with a domain decomposition given by (Lemma 1), i.e $b = 1 - a \approx \sqrt{\epsilon}\log\epsilon^{-1}$. The discretized operator satisfies a maximum principle and we can show that:

$$|e_{0,2}^p| = |e_{N,1}^{p+1}| \geq max_{i=0,\ldots N-1}|e_{i,1}^{p+1}|$$

and

$$|e_{0,2}^p| \geq max_{i=0,\ldots,N}|e_{i,2}^p|.$$

We will call *damping factor* a real ξ such that: $|e_{0,2}^{p+1}| \leq \xi |e_{0,2}^p|$, $\forall p$.

Lemma 2 *Let ϕ_{outer}^p and ϕ_{inner}^p defined by the iterative scheme:*

$$\left\{\begin{array}{l} L^{h_1}\phi_{outer}^p = F \ in \ \Omega_{outer}; \\ \phi_{outer}^p(0) = \alpha_0 \ ; \ \phi_{outer}^p(a) = \phi_{inner}^p(a) \\ L^{h_2}\phi_{inner}^{p+1} = F \ in \ \Omega_{inner}; \\ \phi_{inner}^{p+1}(0) = \alpha_1 \ ; \\ \dfrac{\phi_{inner}^{p+1}(a+h_2) - \phi_{inner}^{p+1}(a)}{h_2} = \dfrac{\phi_{outer}^p(a) - \phi_{outer}^p(a-h_1)}{h_1}. \end{array}\right.$$

Let a be the interface position between the subdomains such that $1 - a \approx \epsilon^{1/2}\log\epsilon^{-1}$. Suppose that $N^{-1} \approx \epsilon^{\frac{1}{2}}\delta$, with $\delta \succ\succ 1$. Then the amplification factor of the iterative scheme is: $\xi \approx \delta^{-1}$.

• **third step: Convergence to the solution of the ODE problem and uniform approximation**

Theorem 1 *Let ϕ be the solution of the Dirichlet problem*

$$L[\phi] = -\epsilon\phi" + \phi = F \ ; \ \phi(0) = \alpha_0, \ \phi(1) = \alpha_1$$

$$Let \qquad \phi^p = \left\{\begin{array}{l} \phi_{outer}^p \ on \ \ M_{outer} \\ \phi_{inner}^p \ on \ \ M_{inner} \end{array}\right.$$

Let $\| \ \|_\infty$ be the maximum norm on the composite grid $M_{outer} \cup M_{inner}$.
Let us suppose that: $N^{-1} \approx \sqrt{\epsilon}\delta, \ ; \ b \sim \sqrt{\epsilon}\log(\epsilon^{-2})$ *with* $\delta \succ\succ 1$
Then

$$\|\phi - \phi^p\|_\infty \leq C(\xi^p + \epsilon\delta)),$$

$$with \qquad \xi \sim \delta^{-1}.$$

Neumann-Dirichlet Scheme

We are going to show that the choice of the boundary conditions at the artificial interface is critical. Let us consider now the F.Q method with Neumann boundary condition at a in Ω_{outer} and the Dirichlet boundary condition at a in Ω_{inner}. The scheme gives:

$$\left\{\begin{array}{l} L^{h_1}\phi^p_{outer} = F \quad \text{on} \quad \Omega_{outer}, \\ \dfrac{\phi^{p+1}_{outer}(a) - \phi^{p+1}_{outer}(a-h_1)}{h_1} = \dfrac{\phi^p_{inner}(a+h_2) - \phi^p_{inner}(a)}{h_2} \quad ; \quad \phi^p_{outer}(0) = \alpha_0 \ , \\ L^{h_2}\phi^p_{inner} = F \quad \text{on} \quad \Omega_{inner}, \\ \phi^p_{inner}(1) = \alpha_1 \quad ; \quad \phi^p_{inner}(a) = \phi^p_{outer}(a). \end{array}\right. \tag{2.4}$$

To start the scheme, we impose an artificial boundary condition at point a. We show that the best choice for the interface position in terms of accuracy is $1-a \approx \sqrt{\epsilon}\log\epsilon^{-1}$ since the formal limit of (2.4) is identical to the formal limit of (2.2). However, this procedure is then highly unstable:

Theorem 2 *Let us assume that $h_1 \succ\succ h_2$ and $h_1 \succ\succ \sqrt{\epsilon}$. Then the amplification factor of the iterative procedure satisfies $\xi \sim \frac{h_1}{h_2}$ and the F.Q procedure with no relaxation is highly unstable.*

With the same principle as below, we proved that we obtain a fast convergence with a good approximation on problems having different operators for each subdomain with the Neumann-Dirichlet scheme. Therefore we found the F.Q. algorithm very interesting for singularly perturbed transmission problems for which the overlapping domain decomposition technique does not be used.

Heterogeneous Domain Decomposition

In this section, we consider a linear second-order transmission problem of the following type:

$$\left\{\begin{array}{l} L_1\phi = -\epsilon\phi" + \phi = F \ in \ \Omega_1 = (0, A); \\ L_2\psi = \psi" = G \ in \ \Omega_2 = (A, 1); \\ \phi(A) = \psi(A); \ \phi'(A) = \psi'(A); \\ \phi'(0) = 0 \ ; \ \psi(1) = 0. \end{array}\right. \tag{2.5}$$

ϵ is a small positive parameter, $\epsilon \in]0, \epsilon_0]$ for some $\epsilon_0 > 0$. In addition we assume the compatibility condition that all derivatives of F vanish in 0. This very simple model is introduced to study the convergence of an heterogeneous domain decomposition based on the F.Q method. We observe that the domain decomposition is dictated by the definition of the transmission problem and that there is no overlap of the subdomains on A.

Asymptotic Analysis

We studied the boundary layer of (2.5) in ([DLTO$^+$96]) and observed that it is a singular perturbation problem with a weak layer of $\sqrt{\epsilon}$ thickness located to the left of A.

First Numerical Procedure

The asymptotic analysis suggests that the computation domain should be split into three subdomains $\Omega_1 = (O, B)$, $\Omega_2 = (B, A)$ and $\Omega_3 = (A, 1)$ where the intermediate subdomain is used to resolve the boundary layer. We assume that F vanishes in the neighbourhood of 0 and that the space step h_i for each subdomain satisfies the asymptotic relation $h_1 \approx \frac{h_2}{A-B} \approx h_3 \approx \frac{1}{N}$. with $b = A - B \prec\prec 1$.

We are going to study the heterogeneous F.Q procedure for such problem. According to the previous analysis, we adopted the F.Q procedure with the D-N boundary conditions to resolve the layer and with the N-D boundary conditions to resolve the transmission problem. The iteration procedure is as follows:

$$\left\{\begin{array}{l}
L_1^{h_1}\phi_1^p = F \ in \ \Omega_1; \\
\phi_1^p(0) = \phi_1^p(h_1) \ ; \ \phi_1^p(B) = \phi_2^p(B); \\
L_2^{h_3}\psi^p = G \ in \ \Omega_3; \\
\psi^p(A) = \phi_2^p(A) \ ; \ \psi^p(1) = 0; \\
L_1^{h_2}\phi_2^{p+1} = F \ in \ \Omega_2; \\
\dfrac{\phi_2^{p+1}(B+h_2) - \phi_2^{p+1}(B)}{h_2} = \dfrac{\phi_1^p(B) - \phi_1^p(B-h_1)}{h_1}; \\
\dfrac{\phi_2^{p+1}(A) - \phi_2^{p+1}(A-h_2)}{h_2} = \dfrac{\psi^p(A+h_3) - \psi^p(A)}{h_3}.
\end{array}\right. \tag{2.6}$$

The proof of convergence of this scheme is very similar to that of the previous section. It can be proved that:

Lemma 3 *Let $(\tilde{\phi}, \tilde{\psi})$ with $\tilde{\phi} = (\tilde{\phi}_{i,j})_{i=0\ldots N, j=1,2}$ and $\tilde{\psi} = (\tilde{\psi}_i)_{i=0\ldots N}$ be the solution of the linear system that is the formal limit of (2.6) when $p \to \infty$. Let M be the composite grid $M = M_1 \cup M_2 \cup M_3$, with*

$$\left\{\begin{array}{l}
M_1 = \{x_{i,1} = i(\frac{B}{N}); i = 0\ldots N\} \\
M_2 = \{x_{i,2} = B + i(\frac{A-B}{N}); i = 0\ldots N\} \\
M_3 = \{x_{i,2} = A + i(\frac{1-A}{N}); i = 0\ldots N\}
\end{array}\right.$$

Let us suppose that $N^{-1} \approx \sqrt{\epsilon}\delta$, with $\delta \succ\succ 1$. Let $\| \ \|_\infty$ be the maximum norm on the composite grid M.

Under the previous hypothesis concerning the discretization and approximation of the operators in each subdomain, $\max(\| \phi - \tilde{\phi} \|_\infty, \| \psi - \tilde{\psi} \|_\infty)$ *is asymptotically minimum when* $b = A - B \approx \sqrt{\epsilon}\log(\epsilon^{-1})$

Proof: see ([DLTO$^+$96])
We have then the following convergence property of the iterative scheme (2.6),

Lemma 4 *Let B be the interface position defined as in Lemma 3. Suppose that $N^{-1} \approx \epsilon^{\frac{1}{2}}\delta$, with $\delta \succ\succ 1$. Then the amplification factor of the iterative scheme is:*

$$\xi \approx \delta^{-1}.$$

Proof: We only need to look at the following homogeneous problem,

$$\left\{\begin{array}{l} L_1^{h_1} e_1^p = 0 \; in \; \Omega_1; \\ e_{1,1}^p = e_{0,1}^p \; ; \; e_{N,1}^p = e_{0,2}^p; \\ L_2^{h_3} e_3^p = 0 \; in \; \Omega_3; \\ e_{0,3}^p = e_{N,2}^p \; ; \; e_{N,3}^p = 0; \\ L_1^{h_2} e_2^{p+1} = 0 \; in \; \Omega_2; \\ \dfrac{e_{1,2}^{p+1} - e_{0,2}^{p+1}}{h_2} = \dfrac{e_{N,1}^p - e_{N-1,1}^p}{h_1}; \\ \dfrac{e_{N,2}^{p+1} - e_{N-1,2}^{p+1}}{h_2} = \dfrac{e_{1,3}^p - e_{0,3}^p}{h_3}. \end{array}\right. \tag{2.7}$$

We obtain for the first subdomain: $e_{i,1}^p = e_{N,1}^p \frac{R^{i-1}+R^{-i}}{R^{N-1}+R^{-N}}, \; \forall i,$

with $R = 1 + \frac{h_1^2}{2\epsilon} + \frac{h_1}{\sqrt{\epsilon}}\sqrt{1 + \frac{h_1^2}{2\epsilon}} \approx \delta^2$. We have then

$$e_{i,1}^p \prec\prec e_{N,1}^p, \; \forall i < N.$$

For the second subdomain, we have $e_{i,3}^p = \frac{N-i}{N} e_{0,3}^p,$ and then $e_{i,N}^p \leq e_{0,3}$.
And for the third subdomain:

$$e_{i,2}^p = h_2 \frac{e_{0,2}^p}{h_1}\left(\frac{R_*^{i-N+1}}{(R_* - 1)(R_*^{-N+1} - R_*^{N-1})} + \frac{R_*^{N-1-i}}{(R_*^{-1} - 1)(R_*^{N-1} - R_*^{-N+1})}\right)$$

$$-h_2 e_{0,3}^p\left(\frac{R_*^{i}}{(R_* - 1)(R_*^{-N+1} - R_*^{N-1})} + \frac{R_*^{-i}}{(R_*^{-1} - 1)(R_*^{N-1} - R_*^{-N+1})}\right),$$

where $R_* = 1 + \frac{h_2^2}{2\epsilon} + \frac{h_2}{\sqrt{\epsilon}}\sqrt{1 + \frac{h_2^2}{2\epsilon}}$.

Using $R_* - 1 \approx \frac{h_2}{\sqrt{\epsilon}}$, we obtain then

$$e_{0,2}^{p+1} \approx \delta^{-1} e_{0,2}^p + 2\sqrt{\epsilon}\exp\left(-\tfrac{b}{\sqrt{\epsilon}}\right) e_{N,2}^p,$$
$$e_{N,2}^{p+1} \approx \sqrt{\epsilon} e_{N,2}^p + 2\sqrt{\epsilon}\exp\left(-\tfrac{b}{\sqrt{\epsilon}}\right)\frac{e_{0,2}^p}{h_1}.$$

We conclude that the amplification factor of the method is then asymptotically δ^{-1}.
Combining Lemma 3 and Lemma 4 we have finally,

Theorem 3 *With the notations defined above, we have:*

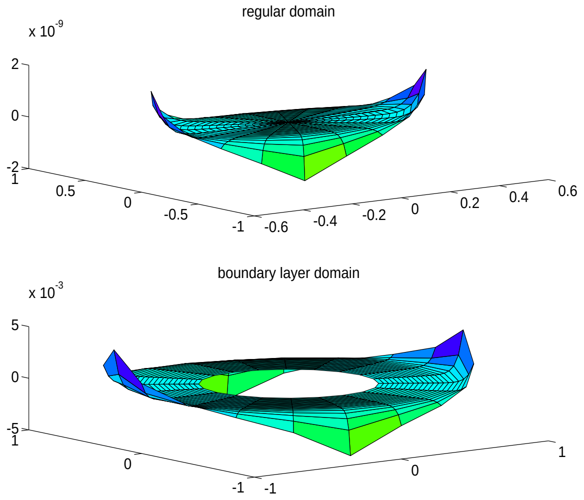
$$\max(\|\phi - \phi^p\|_\infty, \|\psi - \psi^p\|_\infty) \leq C(\xi^p + \epsilon\delta^2), \quad with \quad \xi \sim \delta^{-1}.$$

Proof: The proof is a straightforward application of Lemma 3 and Lemma 4.

Composite Method: Schwarz and F.Q.

Let us use now the *Schwarz alternate procedure* to solve the layer. We keep the F.Q scheme with N - D boundary conditions, solving for the transmission condition in A. We restrict ourselves to an overlap minimum i.e one cell of step h, between $\Omega_1 = [0, a]$ and $\Omega_2 = [b, 1]$ $(with \quad 0 < b < a < 1)$.
Furthermore, to simplify the demonstration, we impose that the grids of the subdomains Ω_1 and Ω_2 coincide at the boundary points.
It can be proved that $\max(\| \phi - \tilde{\phi}\|_\infty, \| \psi - \tilde{\psi}\|_\infty)$ is asymptotically minimum when

Figure 1 solution in metal domain



$$A - b \approx \sqrt{\epsilon}\log(\epsilon^{-1})$$

We have then the following convergence property of the iterative scheme.

Lemma 5 *Let B the interface position defined above. Suppose that $N^{-1} \approx \epsilon^{\frac{1}{2}}\delta$, with $\delta \succ\succ 1$. Then the amplification factor of the iterative scheme is:*

$$\xi \approx \sqrt{\epsilon}\log(\epsilon^{-1})\delta$$

Combining these results, we have finally.

Theorem 4 *With the notations defined above, applying the F.Q and Schwarz mixed method, we have:*
$\max(\|\phi - \phi^p\|_\infty, \|\psi - \psi^p\|_\infty) \leq C(\xi^p + \epsilon\delta^2), \qquad with \qquad \xi \sim \delta\epsilon\log\epsilon^{-1}$

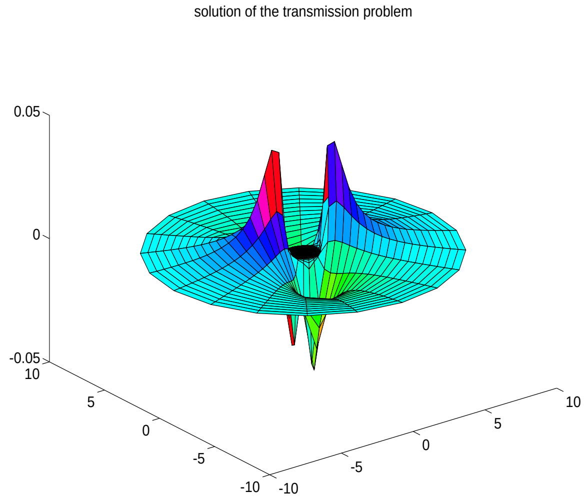
3 Boundary Layers in Two-dimensional Space

Applying some comparison lemmas, we can extend all previous results obtained in a one dimensional space to a two dimensional space with strip domain decomposition. We referred to preprint ([GVC96]) for the detailed of the analysis. We have also applied them to a two dimensional singular perturbed transmission problem that arises in electromagnetic theory [Cou92]. The model is as follows:

$$\begin{cases} -\epsilon\Delta u + a(r,\theta)u = 0 \; on \; \Omega_1 =]0, A[\times[0, 2\pi[, \\ -\Delta u = j(r,\theta) \; on \; \Omega_2 =]A, 1[\times[0, 2\pi[, \\ u_-(A) = u_+(A); u'_-(A) = u'_+(A); \\ u(R_\infty, \theta) = 0 \; for \; \theta \in (0, 2\pi), \end{cases}$$

where Ω_1 is a disk of radius one, Ω_2 is a ring for $r \in (1, R_\infty)$; typically j represents the current density in the inductor, Ω_1 is the domain of the liquid metal, Ω_2 is the

Figure 2 global solution



domain with no conduction, and the boundary layer in Ω_1 corresponds to the well known skin effect. This problem has then been numerically efficiently solved using three subdomains with regular finite difference meshes inside each subdomain, and a very large aspect ratio of the mesh width between the subdomains according to our a priori analysis. The method can then be parallelized at various levels but it is still useless since our test case is a small academic problem. Figure1 shows the solution in domain Ω_1 with the domain decomposition that corresponds to the regular part and the boundary layer. Figure2 shows the global solution. We have also tested our method with finite element discretization and unstructured grids using *modulef* [BPA88]; we keep the radius of the elements per subdomain asymptotically equivalent to the space grid used on the finite difference scheme and find good agreement.

REFERENCES

[BPA88] Bernadou M., PL. G., and Al (1988) *Une bibliothèque modulaire d' éléments finis.* INRIA text book.

[Cou92] Coulaud O. (1992) Asymptotic analysis of magnetic induction with high frequency: solid conductor. Technical Report 3086, INRIA.

[DLTO$^+$96] Desideri J.-A., Le Tallec P., Onate E., Periaux J., and STEIN E. (1996) Analysis of the Funaro-Quarteroni Procedure for Singularly Perturbed Elliptic Boundary Value Problems. In *Numerical Methods in Engineering, ECCOMAS-96*, page 491. John Wiley & Sons,Ltd, Paris.

[FQZ88] Funaro D., Quarteroni A., and Zanolli P. (12 1988) An iterative procedure with interface relaxation for domain decomposition methods. *SIAM J. Num Anal.* 25(6).

[Gar96] Garbey M. (4 1996) A Schwarz Alternating Procedure for Singular Perturbation Problems. *SIAM J. Scient. Comp.* .

[GVC96] Garbey M., Viry L., and Coulaud O. (1996) Analysis of the Funaro-Quarteroni Procedure for Singularly Perturbed Problems. Technical report, Elie-

Cartan Institute.

67

One-level Krylov-Schwarz Domain Decomposition for Finite Volume Advection-diffusion

P. Wilders and G. Fotia

1 Introduction

We consider the two-dimensional advection-diffusion equation discretized with cell-centered finite volumes and the trapezoidal time integration scheme. Several aspects of Krylov-Schwarz domain decomposition will be discussed. The name Krylov-Schwarz refers to methods in which Schwarz domain decomposition is used as a preconditioner for a Krylov subspace method, see the preface of [KX95].

Our interests in domain decomposition are more practical than theoretical. Our goals are towards solving large-scale 'real-life' problems. As such, it is important to work with a method that is not too difficult to implement. One-level Schwarz methods belong to this class. Our attention is drawn by nonoverlapping Schwarz methods (sometimes called Schwarz with minimal overlap), because the use of nonoverlapping subdomains facilitates the implementation. Of course, there is a price to be paid; the number of ddm-iterations will grow if the number of subdomains increases. With a two-level method this might be improved. However, it is not easy to formulate and implement the coarse grid correction. In a parallel environment it is even an open question whether the elapsed time will actually go down if a coarse grid correction is included. Moreover, we are dealing with a time-dependent problem and there is some evidence, see [Cai91], that the behaviour of one-level methods is quite acceptable in this case. Our numerical experiments will confirm this statement to some extent.

In this paper the emphasis is on the description and performance of the domain decomposition iteration. We can only briefly touch upon our experiences with the present method for solving practical problems on parallel computers. We refer the interested reader to [VWMF96] for details on this issue.

The outline of this paper is as follows: Section 2 describes the equations and

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

the discretization. In Section 3 we formulate the interface equations, which establish a reduction of the Schwarz-preconditioned system to a small set of equations involving interface variables only. Section 4 describes our main field of application, single phase tracer flow in a porous medium. In Section 5 we present some numerical results concerning the scalability properties of the ddm-iteration. Finally, Section 6 draws some conclusions and makes some remarks concerning future research.

2 Equations and Discretization

We consider a scalar conservation law of the form

$$’\frac{\partial c}{\partial t} + \nabla .\, [vc - D\nabla c] = 0 \;,\;\; x \in \Omega \in \mathbb{R}^2 \;,\;\; t > 0. \tag{2.1}$$

The coefficients ’, v and D are time-independent. We are interested in the advection-dominated case, i.e. the diffusion tensor D depends on small parameters. For the spatial discretization we employ the cell-centered finite volume method. This leads to the semi-discrete system

$$M\frac{dc}{dt} = Bc. \tag{2.2}$$

M is a diagonal matrix, containing the cell values of the coefficient ’ multiplied with the area of the cell.

B is a sparse matrix with nonzero entries at places defined by the molecule of the discretization; for simple discretizations B is a constant matrix. In order to be able to capture regions with high gradients, advanced nonlinear approximations of the advection terms are often needed, involving sensors and switch functions or limiters. In such a case the entries of B depend on c. We assume that B is a differentiable function of c, which enables us to reach the highest level of time accuracy. We set

$$J = \frac{\partial Bc}{\partial c}. \tag{2.3}$$

For the time-discretization of (2.2) we employ the linearly implicit trapezoidal rule in delta- or incremental formulation, i.e.

$$\left(\frac{M}{\tau_n} - \frac{1}{2}J^n\right)\delta^n = (Bc)^n \;,\;\; \delta^n = c^{n+1} - c^n. \tag{2.4}$$

Here, τ_n denotes the time step. The scheme is second-order accurate in time.

Omitting indices, (2.4) is denoted with

$$A\delta = b \;, \quad A = \frac{D}{\tau_n} - \frac{1}{2}J^n \;, \quad b = (Bc)^n \;. \tag{2.5}$$

The purpose of this paper is to discuss the iterative solution of (2.5) by means of an one-level Krylov-Schwarz domain decomposition method with ILU-preconditioned Bi-CGSTAB for the inversion of the subdomain problems.

3 The Interface Equations

The domain Ω is divided into p nonoverlapping subdomains $\Omega_q, \quad q = 1, ..., p$. The interfaces between the subdomains are an ensemble of some edges of the cells of the finite volume mesh. The nodal points are in the center of the cells (cell-centered finite volumes) and this means that there are no nodal points on the interfaces between subdomains. The set of nodal points is denoted with $z_i, \quad i \in I$. Define the disjunct index sets I_q such that $i \in I_q$ if $z_i \in \Omega_q$. We may partition the matrix A according to these index sets and next define the block Jacobi iteration matrix N. For the sake of simplicity we take $p = 2$ (two subdomains) and in this case there holds

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad N = \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}. \tag{3.6}$$

The block Jacobi matrix N is put forward as the preconditioner, i.e. we consider

$$N^{-1}A\delta = N^{-1}b\,. \tag{3.7}$$

It is well-known, e.g. see [SBG96], that nonoverlaping additive Schwarz and block Jacobi, such as presented above, present identical preconditioners.

In classical substructuring or Schur complement methods the final equations to be solved are formulated in terms of the unknowns associated with nodal points on the interfaces. It is known and used for theoretical purposes that some of the substructuring methods allow for a formulation as a Schwarz method, see [SBG96] and quoted references. It is also possible to go the other way around and to reduce the Schwarz system (3.7) to a smaller set of equations. This has some practical advantages, e.g. the memory requirements (put forward by GMRES) are minimized and the parallel implementation is easier, see [VWMF96]. In the remainder of this section we will briefly describe the main idea behind the reduction, further details may be found in [BW95].

Interface cells are defined as those cells of the finite volume mesh of which the discretization molecule crosses one or more interfaces between subdomains. All other cells are called interior cells. In Figure 1 the interface cells are sketched in the case of a structured quadrilateral finite volume mesh with a 9-point discretization molecule. Figure 2 does the same for an unstructured triangular finite volume mesh with a 10-point discretization molecule. Variables associated with nodal points from the interface cells are called interface variables and all other variables are called interior variables.

Figure 1 Interface cells, 9-point molecule, quadrilaterals.

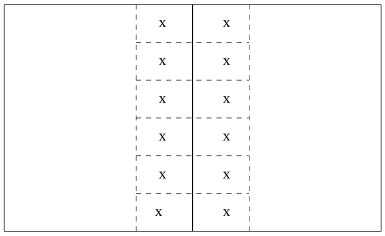
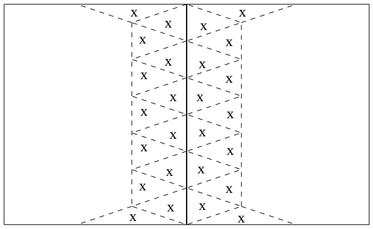


Figure 2 Interface cells, 10-point molecule, triangles.



With α we denote the vector of interface variables, with β the vector of interior variables, and we partition the vector δ of unknowns accordingly, i.e. $\delta = [\alpha,\ \beta]^T$. Let $R = [I\ 0]$ and $Q = [0\ I]$ be the restriction matrices mapping δ to α, respectively δ to β; R^T and Q^T are the corresponding trivial injection matrices.

Now, let δ be arbitrary and set $\alpha = R\delta$, $\beta = Q\delta$. The block Jacobi matrix N and the original matrix A are identical for rows corresponding to interior cells. This means that

$$(N - A)Q^T\beta = 0\ , \quad (N - A)\delta = (N - A)R^T\alpha\ . \tag{3.8}$$

As a consequence (3.7) is equivalent with

$$\begin{bmatrix} P & 0 \\ T & I \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} g \\ h \end{bmatrix}\ , \tag{3.9}$$

with

$$P = RN^{-1}AR^T\ , \quad T = QN^{-1}AR^T\ , \quad g = RN^{-1}b\ , \quad h = QN^{-1}b\ .$$

From (3.9) we see that (3.7) can be reduced to the interface equations

$$P\alpha = g\ . \tag{3.10}$$

GMRES is applied to the reduced system (3.10).

The interface equations (3.10) have been described starting from a basic situation. Generalizations are possible, see [BW95]. Here, we only mention that vertex-centered finite volumes can be treated as well. It suffices to double the unknowns on the interfaces and to augment the system (2.5) similar to [Tan92].

It can be shown that there is equivalence between the interface equations (3.10) and a certain preconditioned Schur system, see [WB95]. The Schur system encountered here is of a more general type than the one encountered in the classical substructuring method. Similar equivalence properties have been established in [BW89], [CG92].

4 Tracer Flow in a Porous Medium

Modeling of two-component single-phase miscible flow is an important issue in reservoir engineering, e.g. see [Ewi83], [Lak89], both per se or as a step towards understanding the numerical properties of more complex multi-phase and multi-component models. The mathematical model describing incompressible miscible flow consists of the transport equation (2.1) for the concentration of the solvent coupled with an elliptic equation for the pressure:

$$-\nabla.(a\nabla p) = 0\ ,\ x \in \Omega\ ,\ t > 0. \tag{4.11}$$

Darcy's law states that the advective velocity v in (2.1) can be obtained from $v = -a\nabla p$.

In the case of tracer flow the coefficient a in (4.11) does not depend on the concentration and/or time and the Darcy velocity is computed only once. Our numerical experiments concern a tracer flow. There holds $a = k/\mu$ with $k(x)$ the permeability of the reservoir and μ the viscosity. In the computations we have used strongly heterogeneous real-life permeability data provided by Agip S.p.A.

In [FQ96] it has been argued that a convenient numerical approximation can be based upon a triangular mixed finite method for (4.11) combined with a cell-centered unstructured finite volume approach for the transport equation (2.1), using the same grid. We employ the linear Brezzi-Douglas-Marini (BDM) element with piecewise constant pressure, e.g. see [BF92]. A variant of the JST-scheme, [JM86], [WFM94], has been adopted for the approximation of the advective terms. The JST-scheme is a central scheme stabilized with a nonlinear artificial dissipation term containing both harmonic and biharmonic operators. The scheme has originally been developed for the solution of nonlinear hyperbolic equations with emphasis on capturing shock fronts and high gradient internal layers with a moderate level of numerical diffusion, see [JST81].

Figure 3 Velocity field.

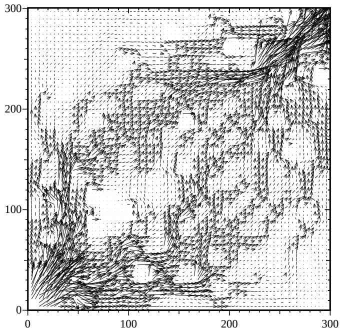
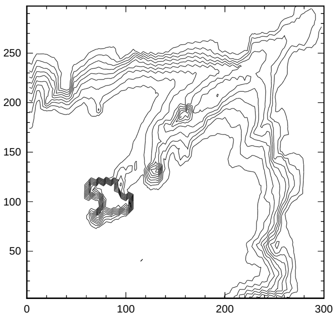


Figure 4 Concentration at .6 PVI.



Our test problem is the quarter of five-spots, a square region with an injection well in the lower left corner and a production well in the upper right corner. In Figure 3 we present the computed velocity field. Figure 4 shows the computed concentration: a front is moving from the injection well to the production well and the wiggles along some horizontal parts of the contour lines are caused by the plotting procedure. A maximal Courant number of $O(40)$ was used in the transport solver. Further physical details can be found in [WFM94]. Both figures illustrate nicely the complexity due to heterogeneity and it shall be clear that fine grids are indispensable in this type of applications.

5 Numerical Results

The square region Ω is divided into p equal sized block shaped subdomains. We fix the number of unknowns N in each subdomain, i.e. $N = 3200$. The total size n of the discrete problem is $n = pN$, i.e. n is growing linearly with the number of subdomains. In a parallel context this is called a memory constrained scaling methodology. Increasing the number of subdomain implies a decreasing spatial grid size and in such a case it is important to take application parameters into account. From the theory of hyperbolic difference schemes it is well-known that the Courant number is the vital similarity parameter and, therefore, we scale such that Courant numbers are fixed. This means that both the spatial grid size and the time-step τ will change if the number of subdomains varies. Domain decomposition methods are not studied often in such a context. Normally, the total size n of the discrete problem is fixed. Nevertheless, questions related to a growing problem size are important if the goal is to use a distributed parallel environment for enlarging the size of the problems treated.

Let us consider a single time step. With M_p we denote the number of matrix-vector multiplications with the matrix R from the interface equation (3.10). There holds $M_p = O_p + 1$, with O_p the number of GMRES ddm-iterations (outer iteration). The subdomain problems are solved iteratively with ILU-preconditioned Bi-CGSTAB (inner iteration). With $I_p(k)$ we denote the number of inner iterations in the $k-$th matrix-vector multiplication and with $\bar{I}_p$ the average number of such iterations, i.e.

$$\bar{I}_p = \frac{1}{M_p} \sum_{k=1}^{M_p} I_p(k).$$

Table 1 presents the measured O_p and $\bar{I}_p$, averaged over the full time interval. The number of ddm-iterations is moderate and increases at an acceptable rate if the number of subdomains grows. It should be remarked, that the overlap parameter H/h is a constant in Table 1 as a consequence of the scaling procedure. ILU-preconditioned Bi-CGSTAB turns out to be very effective, which confirms earlier investigations done in [WFM94].

Table 1 *O_p, the number of outer iterations and $\overline{I}_p$, the averaged number of inner iterations.*

	$p = 4$	$p = 9$	$p = 16$	$p = 25$
O_p	10.1	10.8	12.5	13.1
$\overline{I}_p$	2.6	3.4	3.9	4.5

Let T_1 denote the elapsed time per time step on a sequential computer. The final effect of the numbers found in Table 1 can be measured by means of the normalized sequential time T_1/p. The results can be found in Table 2. The loss of performance, caused by a degrading numerical efficiency, is quite moderate.

Table 2 *T_1/p, the normalized sequential time per time step.*

	p=4	p=9	p=16	p=25
	.63	.71	.87	.93

6 Final Remarks

One-level nonoverlapping Schwarz is one of the easiest to implement domain decomposition preconditioners. We have investigated the method for time-dependent advection-diffusion and the performance turns out to be satisfactory for practical purposes. The number of ddm-iterations, found in Table 1, is still somewhat high. A more advanced preconditioner, like adaptive Robin-Neumann, will certainly lead to an improvement at this point.

Acknowledgement

The research was carried out with financial support of the Dutch Ministry of Economic Affairs and the regional authorities of Sardegna.

REFERENCES

[BF92] Brezzi F. and Fortin M. (1992) *Mixed and hybrid finite element methods.* Springer, Berlin.

[BW89] Bjørstad P. and Wildlund O. (1989) To overlap or not to overlap: a note on a domain decomposition method for elliptic problems. *SIAM J. Sci. Stat. Comput* 10: 1053–1061.

[BW95] Brakkee E. and Wilders P. (1995) The influence of interface conditions on convergence of Krylov-Schwarz domain decomposition for the advection-diffusion equation. Report 95-67, Delft Univ. of Techn., Fac. Techn. Math. and Inf. To be published in J. Scientific Computing.

[Cai91] Cai X. (1991) Additive Schwarz algorithms for parabolic convection-diffusion equations. *Numer. Math* 60: 41–61.

[CG92] Chan T. and Goovaerts D. (1992) On the relationship between overlapping and nonoverlapping domain decomposition methods. *SIAM J. Matrix Anal. Appl.* 13: 663–670.

[Ewi83] Ewing R. (1983) *The mathematics of reservoir simulation.* SIAM, Philadelphia.

[FQ96] Fotia G. and Quarteroni A. (1996) Modelling and simulation of fluid flow in complex porous media. In Kirchgassner K., Mahrenholtz O., and Mennicken R. (eds) *Proc. ICIAM'95.* Akademic Verlag.

[JM86] Jameson A. and Mavripilis D. (1986) Finite volume solution of the two-dimensional Euler equations on a regular triangular grid. *AIAA Journal* 24: 611–618.

[JST81] Jameson A., Schmidt W., and Turkel E. (1981) Numerical solution of the Euler equations by finite volume methods using Runge-Kutta time-stepping schemes. AIAA Paper 81-1259.

[KX95] Keyes D. and Xu J. (eds) (1995) *Proc. of the Seventh International Symposium on Domain Decomposition methods in Science and Engineering.* AMS, Providence.

[Lak89] Lake L. (1989) *Enhanced Oil Recovery.* Prentice Hall, New Jersey.

[SBG96] Smith B., Bjørstad P., and Gropp W. (1996) *Domain Decomposition; parallel multilevel methods for elliptic partial differential equations.* Cambridge University Press, Cambridge, UK.

[Tan92] Tang W. (1992) Generalized Schwarz splittings. *SIAM J. Sci. Stat. Comput* 13: 573–595.

[VWMF96] Vittoli C., Wilders P., Manzini M., and Fotia G. (1996) Distributed parallel computation of 2D miscible transport with multi-domain implicit time integration. Report 96/50, CRS4, Cagliari, Italy. To be published in J. Simulation Practice and Theory.

[WB95] Wilders P. and Brakkee E. (1995) Schwarz and Schur: a note on finite volume domain decomposition for advection-diffusion. Report 95-59, Delft Univ. of Techn., Fac. of Techn. Math. and Inf.

[WFM94] Wilders P., Fotia G., and Marrone M. (1994) Implicit time stepping and domain decomposition for 2D miscible flow in porous media with unstructured finite volumes. Report CRS4 AppMath-94-23, CRS4, Cagliari, Italy.

68

Optimization of Flexible Coupling in Domain Decomposition for a System of PDEs

H. T. M. Van Der Maarel and A. W. Platschorre

1 Introduction

Domain decomposition (DD) may be applied to boundary-value problems for various reasons, ranging from the wish to solve the discretized problem on a (massively) parallel machine or a cluster of scalar machines, to the necessity to use different mathematical or numerical models in different parts of the domain of definition, or the need of a flexible modeling technique on complex domains. In any case, the performance of domain decomposition methods is of utmost importance. For an application running on a parallel architecture, computing time may be duly saved, in spite of the overhead imposed by the domain decomposition method. The performance of a DD method becomes especially a factor of importance when such a method is to be used in a non-parallel environment. In that case there is no gain in wall-clock time for the method, which could compensate for the method's overhead. Then, the benefits of possible gain in modeling-flexibility or possibly a higher accuracy of results obtained with a DD method, must compensate for the DD method's overhead cost.

In the present paper we report on the optimization of a flexible coupling technique for a system of PDEs.

Optimization of Interface Conditions

We consider a two-grid DD method for a system of partial differential equations. For our analysis the domain of definition is divided into two disjoint subdomains, on each of which a subproblem is defined. The subproblems are artificially decoupled. The coupling between the subproblems, such that the substructured problem becomes equivalent to the original problem, is restored in the iteration scheme applied on the level of the subproblems. The convergence of this iteration scheme and hence the performance of the DD method, depends strongly on the equations (interface conditions) that are used to restore the coupling of the subproblems. The proposed

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

method features the optimization of a set of parameters which appear in the interface conditions. Optimization is considered w.r.t. the convergence rate of the additive Schwarz method used in the subproblem iteration scheme.

The Generalized Schwarz Coupling and Convergence

A flexible coupling mechanism is obtained by introducing a set of free parameters in the interface conditions. An optimal set of values for the parameters in a given problem is obtained when the best convergence rate of the subproblem-iteration, over all possible values that these parameters can take, is achieved. The optimal values depend on the specific problem at hand, as well as on the choices made for the substructuring, the discretization used to obtain a set of algebraic equations and the iteration scheme on the level of subproblems to solve this set of equations.

Our starting point is the optimization of a flexible coupling technique proposed by Tan and Borsboom [TB93] and Tan [Tan95]. This method is based on a generalization of the classical Schwarz algorithm, known as 'Generalized Schwarz Splitting' by Tang [Tan92]. For a one-dimensional two-point boundary-value problem Tang obtained an increase in convergence rate, with the asymptotic convergence factor changing from 0.91 for the classical Alternating Schwarz Method (requiring 60 iterations to satisfy his convergence criterion) to 10^{-4} (requiring only 3 iterations to satisfy his criterion) for Tang's generalized Schwarz method. Tan and Borsboom applied Tang's generalized Schwarz to a two-dimensional advection-diffusion problem and obtained an asymptotic convergence factor of 0.3. Their even-more-generalized Schwarz method, with interface conditions including second-order cross-derivatives, gives an asymptotic convergence factor of 0.05. The latter method is the generalized Schwarz method that we adopt as a starting point for our method and which we will extend for use with a set of PDEs.

We consider a DD method for a system of n linear partial differential equations, discretized with a finite-difference scheme. The domain of definition Ω is divided in two parts denoted by Ω_i, $i = 1, 2$, with a common boundary Γ. For the interface conditions on iteration level m of the subproblem defined on Ω_i, we use a discretization of the general interface condition

$$u_i^{(m)} + \alpha \frac{\partial u_i^{(m)}}{\partial n} + \beta \frac{\partial u_i^{(m)}}{\partial t} + \gamma \frac{\partial^2 u_i^{(m)}}{\partial t \partial n} = \\ u_i^{(m-1)} + \alpha \frac{\partial u_i^{(m-1)}}{\partial n} + \beta \frac{\partial u_i^{(m-1)}}{\partial t} + \gamma \frac{\partial^2 u_i^{(m-1)}}{\partial t \partial n}.$$

Here n and t denote normal and tangential directions on Γ, respectively. The discretized interface conditions involve the values the unknown function variables across the interface Γ.

A local mode analysis is applied to reveal the relation between the interface parameters α, β, and γ on the one hand and the asymptotic convergence rate of the iteration process on the other hand. From this analysis the sensitivity of the convergence rate as depending on the interface parameters can be studied, which seems to be an important issue for the practical application of the proposed DD method.

Finally an optimization algorithm is applied, which is used to obtain an optimal set of values for the parameters in each of the interface equations. Unfortunately, it appears that for the cases considered, the optimal set of interface parameters and the corresponding asymptotic convergence factors are very close to the classical Dirichlet-Dirichlet domain coupling of Schwarz' original method.

2 Model Equations and Discretization

The area of application that we will concentrate on in the future, is part of the field of viscous CFD for ship hydrodynamics. Therefore, our target is the incompressible, steady Navier-Stokes equations.

The Reduced Navier-Stokes Equations

The method for the Navier-Stokes equations that we consider is based on a finite-difference discretization of the steady equations in generalized coordinates. This method features the neglect of diffusion in the 'main stream' direction (parabolization) and a *downwind* discretization of the pressure derivative in this direction. The result is called *partially parabolized* or *reduced* discretization of the Navier-Stokes equations. In a Cartesian coordinate system (x, y) in R^2 and for $(x, y) \in \Omega \subset R^2$, they are given by

$$\begin{aligned} \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0, \\ u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{\partial p}{\partial x} - D\frac{\partial^2 u}{\partial y^2} &= 0, \\ u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{\partial p}{\partial y} - D\frac{\partial^2 v}{\partial y^2} &= 0, \end{aligned}$$

where $u : \Omega \to R$ and $v : \Omega \to R$ are Cartesian velocity components in x and y direction, respectively and where $p : \Omega \to R$ denotes the (generalized) pressure. The constant $D > 0$ is the diffusion coefficient. The above set of equations is supplemented with an appropriate set of boundary conditions.

In this paper we consider a model for the reduced Navier-Stokes equations. This model will be used in a local mode analysis for a domain-decomposition method which requires the model to be linear. Since the discretization of the pressure gradient plays an important role in our Navier-Stokes method, we will not adhere to the usual set of convection-diffusion equations as a model for analysis. The linearization of the reduced Navier-Stokes equations that we use includes convection and diffusion of momentum, driven by a *known* convection field with a constant diffusion coefficient and driven by the unknown pressure gradient. Furthermore, the continuity equation is kept in the model. From the physical point of view this equation will act as a constraint on the pressure field. The model equations considered are then given by

$$\begin{aligned}
\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y} &= 0,\\
a\frac{\partial u}{\partial x}+b\frac{\partial u}{\partial y}+\frac{\partial p}{\partial x}-D\frac{\partial^2 u}{\partial y^2} &= 0,\\
a\frac{\partial v}{\partial x}+b\frac{\partial v}{\partial y}+\frac{\partial p}{\partial y}-D\frac{\partial^2 v}{\partial y^2} &= 0,
\end{aligned}$$

again supplemented with an appropriate set of boundary conditions. Here, $a:\Omega\rightarrow R$ and $b:\Omega\rightarrow R$ are the Cartesian components of the convection field in x and y direction, respectively. Without loss of generality we will assume $a,b>0$. The main stream direction is chosen to be in the x direction, which leads to the property $a>b$. For physical relevance, we will assume that (a,b) is divergence-free. The domain is assumed to be subdivided in two subdomains, with a common boundary Γ. In his paper Γ is assumed to be perpendicular to the main stream direction x.

The Discretized Equations

The discretization that we use is a standard finite-difference discretization. In this paper the first derivatives of velocity components are discretized with the first-order upwind finite-difference formula. Second-order derivatives are discretized with standard second-order finite-differences. The pressure derivative is discretized with the first-order accurate, downwind finite-difference. See Hoekstra [Hoe92] for more information on the particular choice for the discretization.

The domain Ω is divided in two domains Ω_i, $i=1,2$. A discrete approximation of a function $u_k:\Omega_k\rightarrow R$, $k=1,2$, is denoted by $u_k^{i,j}=u_k(i\Delta x, j\Delta y)$, $0\le i\le I_k+1$ and $0\le j\le J+1$. Here, Δx and Δy denote constant step sizes in x and y direction, respectively. Following this scheme, the discretized equations are given by

$$\begin{aligned}
\frac{u_k^{i,j}-u_k^{i-1,j}}{\Delta x}+\frac{v_k^{i,j}-v_k^{i,j-1}}{\Delta y} &= 0,\\
a\frac{u_k^{i,j}-u_k^{i-1,j}}{\Delta x}+b\frac{u_k^{i,j}-u_k^{i,j-1}}{\Delta y}+\frac{p_k^{i+1,j}-p_k^{i,j}}{\Delta x}-D\frac{u_k^{i,j+1}-2u_k^{i,j}+u_k^{i,j-1}}{\Delta y^2} &= 0,\\
a\frac{v_k^{i,j}-v_k^{i-1,j}}{\Delta x}+b\frac{v_k^{i,j}-v_k^{i,j-1}}{\Delta y}+\frac{p_k^{i,j+1}-p_k^{i,j}}{\Delta y}-D\frac{v_k^{i,j+1}-2v_k^{i,j}+v_k^{i,j-1}}{\Delta y^2} &= 0.
\end{aligned}$$

The general interface conditions are discretized using central finite differences. Following Tan and Borsboom in [TB93], the coefficient γ for the cross-derivative terms in the interface conditions is taken equal to $\beta\Delta x$, thus reducing the number of free parameters, while we may still be confident to obtain a considerable increase in convergence rate for an optimal set of parameters. For the model problem the interface equations on Γ are given by

$$\begin{aligned}
&\frac{1}{2}(u_2^{1,j}+u_2^{0,j})+\frac{\alpha_1}{\Delta x}(u_2^{1,j}-u_2^{0,j})+\frac{\beta_1}{2\Delta y}(u_2^{0,j+1}-u_2^{0,j-1})= \\
&\qquad \frac{1}{2}(u_1^{I_1+1,j}+u_1^{I_1,j})+\frac{\alpha_1}{\Delta x}(u_1^{I_1+1,j}-u_1^{I_1,j})+\frac{\beta_1}{2\Delta y}(u_1^{I_1,j+1}-u_1^{I_1,j-1}), \\
&\frac{1}{2}(v_2^{1,j}+v_2^{0,j})+\frac{\alpha_2}{\Delta x}(v_2^{1,j}-v_2^{0,j})+\frac{\beta_2}{2\Delta y}(v_2^{0,j+1}-v_2^{0,j-1})= \\
&\qquad \frac{1}{2}(v_1^{I_1+1,j}+v_1^{I_1,j})+\frac{\alpha_2}{\Delta x}(v_1^{I_1+1,j}-v_1^{I_1,j})+\frac{\beta_2}{2\Delta y}(v_1^{I_1,j+1}-v_1^{I_1,j-1}), \\
&\frac{1}{2}(p_1^{I_1+1,j}+p_1^{I_1,j})+\frac{\alpha_3}{\Delta x}(p_1^{I_1+1,j}-p_1^{I_1,j})+\frac{\beta_3}{2\Delta y}(p_1^{I_1+1,j+1}-p_1^{I_1+1,j-1})= \\
&\qquad \frac{1}{2}(p_2^{1,j}+p_2^{0,j})+\frac{\alpha_3}{\Delta x}(p_2^{1,j}-p_2^{0,j})+\frac{\beta_3}{2\Delta y}(p_2^{1,j+1}-p_2^{1,j-1}).
\end{aligned}$$

3 Fourier Analysis

For the set of algebraic equations defined above and an additive Schwarz iteration scheme, we perform a local mode analysis. Therefore, we supplement the set of equations with Dirichlet boundary conditions on the inlet and outlet parts of the domain boundary.

Denote an approximation after m iterations with $u_k^{(m)} = \bigcup_{i,j} u_k^{i,j(m)}$. Similar notations will be used for other grid functions. We assume solutions which are periodic in y. Discrete Fourier transformation in the direction along the interface Γ gives for the error components after the mth iteration

$$\begin{aligned}
e_k^{i,j(m)} &= u_k^{i,j(m)} - u_k^{j,k(*)} &&= \sum_{s=0}^{J-1} \rho_{i,s,k}^{(m)} \mathrm{e}^{\mathrm{i}j\theta_s}, \\
f_k^{i,j(m)} &= v_k^{i,j(m)} - v_k^{j,k(*)} &&= \sum_{s=0}^{J-1} \sigma_{i,s,k}^{(m)} \mathrm{e}^{\mathrm{i}j\theta_s}, \\
g_k^{i,j(m)} &= p_k^{i,j(m)} - p_k^{j,k(*)} &&= \sum_{s=0}^{J-1} \tau_{i,s,k}^{(m)} \mathrm{e}^{\mathrm{i}j\theta_s},
\end{aligned}$$

where $\theta_s = \frac{2\pi s}{J}$, $s = 0, \ldots, J-1$ and $\rho_{i,s,k}$, $\sigma_{i,s,k}$ and $\tau_{i,s,k}$ are the discrete Fourier transforms. The superscript $(*)$ indicates exact solution of the algebraic equations.

For each Fourier mode s the transformed equations for the Fourier transforms $\rho_{i,s,k}$, $\sigma_{i,s,k}$ and $\tau_{i,s,k}$ of the errors $e_k^{i,j}$, $f_k^{i,j}$ and $g_k^{i,j}$ can be written as

$$\begin{pmatrix} \rho_i \\ \sigma_i \\ \tilde{\tau}_i \end{pmatrix}_{s,k} = \begin{pmatrix} 1 & -a\frac{H_s}{G_s}\mathrm{e}^{-\mathrm{i}\theta_s} & \frac{H_s^2}{G_s}\mathrm{e}^{-\mathrm{i}\theta_s} \\ 0 & \frac{a}{G_s} & -\frac{H_s}{G_s} \\ a-G_s & aH_s\mathrm{e}^{-\mathrm{i}\theta_s} & 1-H_s^2\mathrm{e}^{-\mathrm{i}\theta_s} \end{pmatrix} \begin{pmatrix} \rho_{i-1} \\ \sigma_{i-1} \\ \tilde{\tau}_{i-1} \end{pmatrix}_{s,k},$$

where

$$G_s = a + b\frac{\Delta x}{\Delta y}(1 - \mathrm{e}^{-\mathrm{i}\theta_s}) - 2D\frac{\Delta x}{\Delta y^2}(\cos\theta_s - 1),$$

$$H_s = \frac{\Delta x}{\Delta y}(\mathrm{e}^{\mathrm{i}\theta_s} - 1),$$

and where we used $\tilde{\tau}_i = \tau_{i+1}$, $i = 0, \dots, I_k$. The above recurrence relation has a general solution, which can be written as

$$\begin{pmatrix} \rho_i \\ \sigma_i \\ \tilde{\tau}_i \end{pmatrix}_{k,s} = R_{k,s}\lambda_{s,1}^i e_1 + S_{k,s}\lambda_{s,2}^i e_2 + T_{k,s}\lambda_{s,3}^i e_3.$$

Here $\lambda_{s,n}$ denotes the nth eigenvalue of the recurrence matrix, with a corresponding eigenvector e_n, $n = 1, 2, 3$.

If we now make Fourier transforms of the boundary conditions and interface equations, the iteration index m for the additive Schwarz iteration enters the equations, and we can readily derive an iteration equation for $z_s^{(m)} = (R_{1,s}^{(m)}, S_{2,s}^{(m)}, T_{2,s}^{(m)})^T$. Here the derived algebraic relations between $S_{1,s}$, $T_{1,s}$ and $R_{1,s}$ and between $R_{2,s}$ and $S_{2,s}$ and $T_{2,s}$ are used to eliminate $S_{1,s}$, $T_{1,s}$ and $R_{2,s}$. The iteration can be written as

$$Mz^{(m)} = Nz^{(m-1)}.$$

The 3×3 matrix M has a very simple structure and can be easily inverted analytically. The eigenvalues $\chi_i(M^{-1}N)$, $i = 1, 2, 3$, of the iteration matrix $M^{-1}N$, the spectral radius $\rho(M^{-1}N) = \max_i |\chi_i(M^{-1}N)|$ and asymptotic convergence factor $\rho_\infty(M^{-1}N) = \max_{\theta_s} \rho(M^{-1}N)$ can be determined. The asymptotic convergence factor depends on the free parameters in the interface conditions.

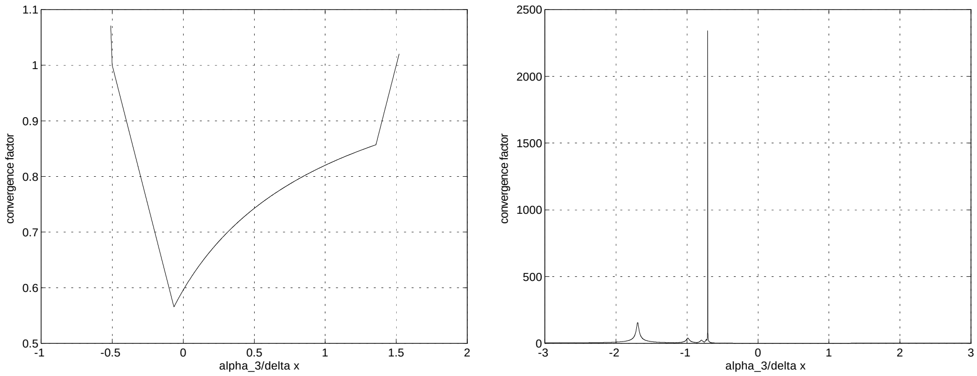
4 Optimization of the Interface Conditions

The asymptotic convergence rate as a function of the free parameters in the interface conditions can be studied. In order to choose an optimization algorithm to find the best possible asymptotic convergence rates, a better understanding of and insight in the asymptotic convergence rate as a function of the free parameters is a prerequisite.

Convergence Factor

The convergence factor is the maximum over all Fourier modes and the three eigenvalues of the iteration matrix. While changing the free parameters, the asymptotic convergence factor may be reached at a different eigenvalue and/or at a different Fourier mode. So, even if the eigenvalues for a fixed Fourier mode depend C^1 continuously on the free parameters, ρ_∞ is not necessarily a C^1 continuous function of the parameters. The discrete set of Fourier modes may be extended to include all modes $0 < \theta < 2\pi$. Then, in general, the asymptotic convergence factor as a function of the free parameters will only be non-differentiable at the intersection points $\chi_i = \chi_j$, $i \neq j$. Therefore, any technique used for the optimization of the free parameters in the interface conditions with respect to convergence factor should not require the derivative of the asymptotic convergence factor with respect to the free parameters. An example of the convergence factor as a function of a free parameter is given in Figure 1. Here the coefficient of the x component of the pressure gradient is varied, while the coefficients are fixed (at there optimal values). It appears that the convergence factor is quite strongly dependent on the interface parameters, which makes a sensible choice for these parameters even more important.

Figure 1 Asymptotic convergence factor as a function of $\alpha_3/\Delta x$.



Optimization

The optimal values of the free parameters can be computed by applying an appropriate minimization algorithm. However, any of the classical minimization schemes will fail in general, unless the starting value is chosen sufficiently close to the solution. This is particularly so for the present case, since the asymptotic convergence factor as a function of the free parameters appears to have a number of local minima.

We apply a minimization algorithm by Powell [Pow64], which is based on a one-dimensional or line minimization algorithm. The line minimizations are consecutively applied in mutually conjugate directions, which are constructed during the minimization process. As a line minimization an inverse parabolic interpolation scheme is used.

As an example, we present in Table 1 the results of a numerical optimization of the free parameters and their corresponding (optimal) convergence factors for the model problem, as predicted by the Fourier analysis presented in this paper. We considered

Table 1 Optimal parameters and convergence factors.

I	$\alpha_1/\Delta x$	$\beta_1/\Delta y$	$\alpha_2/\Delta x$	$\beta_2/\Delta y$	$\alpha_3/\Delta x$	$\beta_3/\Delta y$	ρ_∞	ρ_∞^c
10	-0.3941	0.0986	-0.2022	10.67	-0.0672	-0.0012	0.567	0.594
100	-0.0799	0.2937	0.0607	17.47	-0.0607	0.0576	0.561	0.594

a domain $\Omega = (0,2) \times (0,1)$ and used a constant convection field with $\frac{b}{a} = 0.1$, Péclet number $\mathrm{Pe} = \frac{a\Delta x}{D} = 10^4$ and number of steps in i direction $I_1 = I_2 = I$. The table also contains the asymptotic convergence factor ρ_∞^c for the classical Schwarz method, as predicted with the Fourier analysis. Unfortunately, it appears that the optimal value of the asymptotic convergence factor is not significantly different from that of the classical Schwarz method. In an experiment which numerically solves the discretized problem in the present DD context, using the optimal interface parameters from Table 1, the convergence factor was found to be 0.60. This is in good agreement with the theoretical convergence factors presented in Table 1.

5 Concluding Remarks

The DD method presented involves optimization of interface conditions with respect to the convergence rate of the additive Schwarz iteration scheme used on the level of the subproblem blocks. As such, the method may be considered a generalized Schwarz splitting method. We considered a first-order accurate discretization of a model for the reduced Navier-Stokes equations and an interface perpendicular to the main stream direction. The interface equations involve the unknown function values, first-order derivatives in normal and tangential direction and the second-order cross-derivative term.

For the rather simple discretization used, a Fourier analysis for the additive Schwarz iteration can be done analytically with the aid of modern tools from computer algebra.

The Fourier analysis revealed the relation between the asymptotic convergence rate and the free parameters in the interface conditions.

In general, the asymptotic convergence rate is a non-smooth function of the free parameters. The minimization of Powell may be used to obtain a local minimum in the asymptotic convergence factor as a function of the parameters. However, this function may have a substantial number of local minima, and such a minimization method is not suited for general application.

Unfortunately, the minimum convergence factor for the generalized interface conditions as obtained by the optimization, is only slightly smaller than the convergence factor for the classical Dirichlet-Dirichlet coupling of the original Schwarz method. This is contrary to the expectations, based on results obtained by Tang [Tan92] and Tan and Borsboom [TB93].

The reason for this may be the one-sided upwind and downwind discretization of the various terms in the PDEs considered. Furthermore, early in our analysis we decided to make the coefficient of the cross-derivative term in the interface conditions equal to the coefficient of the tangential-derivative term in the interface conditions. This choice was based on the results obtained by Tan and Borsboom, who with the same choice for a scalar advection-diffusion equation have been rather successful. For the present case this choice may be less appropriate.

The minimum convergence factor seems to be quite sensitive to the choice of the parameters in the interface condition. However, further research has shown that, in the case considered and the specific choices that we made, the optimal convergence rate itself is not very sensitive for problem parameters such as convection angle, the mesh Péclet number or the number of grid cells used.

REFERENCES

[Hoe92] Hoekstra M. (1992) Some fundamental aspects of the computation of incompressible flows. In *Proc. of the Second Osaka International Colloquium on Viscous Fluid Dynamics in Ship and Ocean Technology*, pages 158–170.

[Pow64] Powell M. (1964) An efficient method for finding the minimum of a function of several variables, without calculating derivatives. *Computer J.* 7: 155–162.

[Tan92] Tang W. (1992) Generalized Schwarz splittings. *SIAM J. Sci. Stat. Comput.* 13(2): 573–595.

[Tan95] Tan K. (1995) *Local Coupling in Domain Decomposition.* PhD thesis, University Utrecht, Department of Mathematics.

[TB93] Tan K. and Borsboom M. (1993) Problem-dependent optimization of flexible couplings in domain decomposition methods, with an application to advection dominated problems. Research report, Delft Hydraulics, Delft, The Netherlands.

69

Intelligent Interfaces of a Schwarz Domain Decomposition Method via Genetic Algorithms for Solving Nonlinear PDEs: Application to Transonic Flows Simulations

Jacques Periaux, Bertrand Mantel and Hong Quan Chen

1 Introduction

Genetic Algorithms (GAs) mimic natural selection based on the Darwinian *Survival of the Fittest* principle. These evolution algorithms inspired from biology share digital information and have been introduced by J.H. Holland [Hol92]. A fitness function is chosen to classify individuals of a population in terms of their adaptation to their environment. Those individuals are candidate solutions of a minimization problem and have a digital DNA representation by binary strings. The robustness of Genetic Algorithms rely essentially on genetic recombination involving selection, crossover, and mutation random operators. They are able to explore very large search spaces to find near-global minima whilst traditional methods with gradient information may fail.

An implementation of GAs for the solution of nonlinear flow problems using domain decomposition methods is presented. It is shown that the conventional Schwarz iterative methods for the matching of overlapped subdomain solutions can be extended to nonlinear situations with a genetic treatment at the interfaces. The fitness function considered in this problem is the distance of local solutions on the overlapping regions. Intelligent interfaces act by disseminating genetic information at one node located on the interface to neighbors for appropriate boundary conditions. Combining the domain decomposition method with the evolution process, converged genetic solutions of transonic shocked flows in nozzles and around lifting multi airfoils are computed successfully.

A particular emphasis is given to the effect of discretization on convergence speed and parallel properties of the evolution method. The available results show that the new method based on local niching strategy has the potential to remove

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

the dependence of mesh size without a preconditioner and can be also very easily implemented on a distributed parallel computer due to its inherent parallelism.

2 Brief Description of Genetic Algorithms

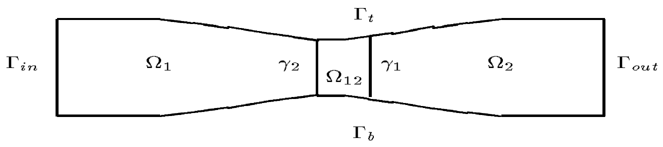
In order to construct implementations of GAs for the solution of nonlinear flow problems using domain decomposition methods, we will briefly restate some of the principle of optimization available in open literature using GAs. Consider a parameter optimization problem of minimizing the cost index $J = f(x)$, where the parameter is x . The first step of optimization process is to code the parameter x as a finite-length string because GAs work with a coding of the parameter set. Here we take a binary string, say of length 8 , such that the lower bound X_{min} for the variable maps to 00000000 and the upper bound X_{max} maps to 11111111 , with a linear mapping in between. The length of string is governed by the required accuracy of the solution. Keep in mind that in general, the string might represent a possible solution to the problem parameterized.

With the described coding, a population of randomly generated strings, say of population size N, would be used as a starting point of the GAs. The strings are then decoded and evaluated to obtain a quantitative measure of how well (fitness values) they behave as possible problem solutions, and transformed to form next generation of size N by using GA-operators. This process continues for successive generations until convergence is achieved or a near optimal solution is found. We can see that the operators play the leading role in the whole processing of the GAs.

There are many existing GA-operators. Among them, three operators: reproduction, crossover, and mutation (which is widely known as the simplest form of a genetic algorithm, namely SGA [Gol89]), are used for many practical uses. Reproduction is a process by which strings with better fitness values receive correspondingly large numbers of copies in the following generation, which is similar to the concept of "survival of the fittest" in the Darwinian theory. The reproduction operator can be implemented artificially in a number of ways, such as Roulette Wheel Selection. In this paper we use Tournament Selection[Gol89] , a simple method of implementing reproduction, to fill up the mating pool where newly reproduced strings are placed and await the action of the other operators.

After reproduction, the operator crossover follows in three steps. First, two newly produced strings are randomly chosen as parents from the mating pool. Second, a position along the two strings is selected uniformly at random. Finally, based on a probability crossover P_c, the paired strings exchange all characters following the crossing site. Clearly, the crossover propagates a structured random information exchange between better fitted parents to produce two offsprings which are expected to combine the better fitted characters of their parents. The operator that merely causes a random alteration of a string position based on probability P_m is called mutation. In present case, this involves changing a bit 1 to a 0 and vice versa. In general, the mutation operator improves the population diversity and prevents the convergence to local minima. For a more thorough discussion of the theoretical foundations of the GAs see the works of J. H. Holland [Hol92] and D. E. Goldberg [Gol89].

Figure 1 Description of a nozzle with two subdomains



3 Implementations of GAs for the Domain Decomposition Problem

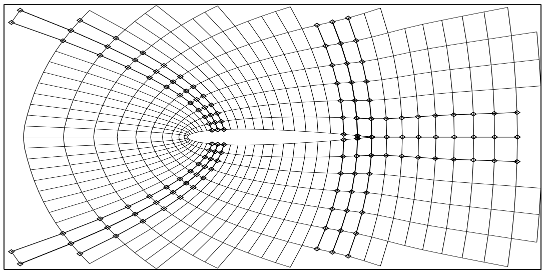
Description of the Problem

The problems of transonic flows in a nozzle and particularly the flows around a lifting airfoil will be investigated in this paper. In order to present general idea of implementations of genetic algorithms for the domain decomposition problem, we will first describe the problem based on the flows in a simple nozzle for sake of simplicity. As shown in Figure 1 , we decompose the computational domain Ω in two subdomains Ω_1 and Ω_2 with overlapping Ω_{12} whose interfaces are denoted by γ_1 and γ_2 . We shall take values, g_1 on γ_1 and g_2 on γ_2, as extra boundary conditions in order to obtain solution in each subdomain. Using domain decomposition techniques, the problem can be reduced to minimize the following function:

$$J(\widetilde{g_1}, g_2) = \frac{1}{2} \parallel \Phi_1 - \Phi_2 \parallel^2, \tag{3.1}$$

where Φ_1 and Φ_2 are the solutions in the overlapping subdomain Ω_{12} , $\parallel \bullet \parallel$ denotes an appropriate norm and $\widetilde{g_1}$ the decoded values of genetic g_1 representation. In the following sections we will present new derivative-free methods of optimization to minimize the above function J based on genetic algorithms.

Figure 2 Subdomains with marked overlappings (mesh size: 87 × 15)



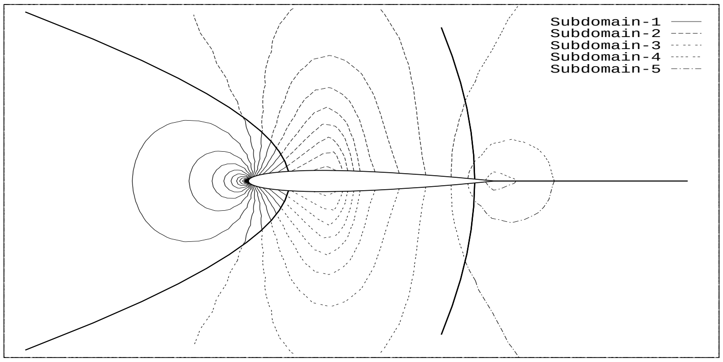
Implementation 1 without niching

For the function being optimized, the variables are rewritten in a code to form a structured string that GAs can directly operate on. For the problem described above, binary codings for multiparameters are used, and we only code g_1 , which can be

$$\widetilde{g_{1i}}; \quad i = 1, N \qquad (N \ \ parameters)$$

each $\widetilde{g_{1i}}$ is coded in l_i bits, thus the length of the string can be $L = \sum_i^n l_i$.

Figure 3 Iso-Mach lines in each subdomain , Mach= 0.8, Attack= 0.0^o



Let us consider population size 25 (i.e. 25 strings). GAs decode each string to return the values of g_{1i} . With g_{1i} known, we can compute the solutions of the domain Ω_1, namely Φ_1. Like Schwarz's alternative method, we take g_2 based on Φ_1, (i.e. $g_2 = \Phi_1 \mid_{\gamma_2})$, thus the solution , Φ_2 , in the domain Ω_2 can be calculated. Now, we send both values in overlapping to cost function:

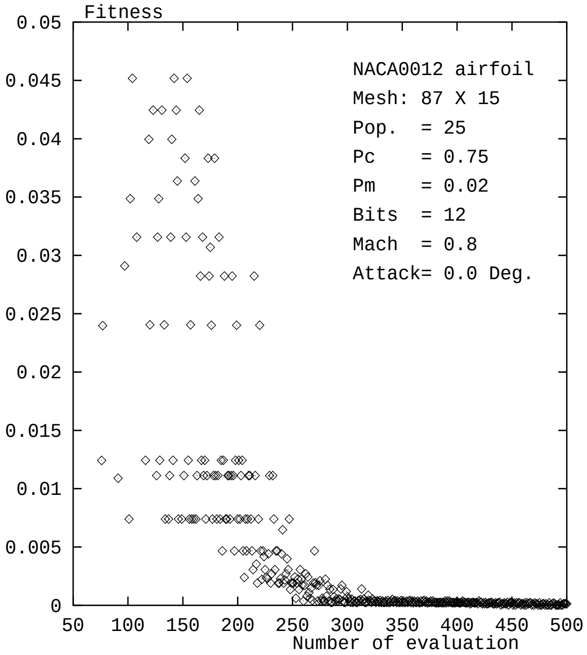
$$J(\widetilde{g_1}) = \frac{1}{2} \parallel \Phi_1 - \Phi_2 \parallel^2 . \tag{3.2}$$

Thus , each string has cost value. New generation of strings can be produced by performing GA-operators and applying survival of the fittest principle.

Implementation 2 with niching

Similar to the Implementation 1 of GAs, we make parameters in each overlapping subdomain belong to one local niche. In each local niche, choose one or several parameters as important parameters. All important parameters are then coded but

Figure 4 Non-lifting case , Mach= 0.8, Attack= 0.0^o



other parameters are nested or coded in a nested way. This could mean that N parameters (g_{1i} , i=1,N) belong to one local niche. We choose g_{11} as key parameter, which is coded in l bits, and other parameters are nested to the decoded value, $\widetilde{g_{11}}$. The nested values can be calculated

$$\Delta_i = g_{1i} - \widetilde{g_{11}} \tag{3.3}$$

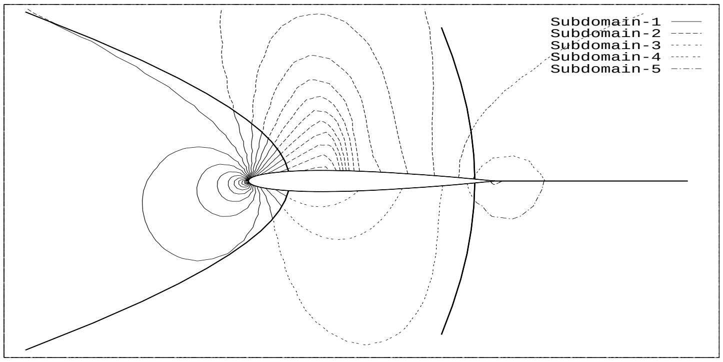
each Δ_i can be coded now in l_i bits. Keep in mind that $\Delta_i(i = 2, N)$ are nested values and the length of $l_i(i \geq\ 2)$ can be short as compared with that for key parameter. This means that we can use local niching strategy to keep the proper representation space, which may reduce the effect of the size of discretization.

In this paper we predict Δ_i based on numerical information, which can be

$$\Delta_i^{n+1} = g_{1i}^n - \widetilde{g_{11}^n} \tag{3.4}$$

where the superscript n is corresponding to the n-th generation and g_{1i}^n are obtained from the previous solution Φ_2 in the domain Ω_2. For simplicity the values of Δ_i can be kept the same in one generation and can be updated with the fittest individual of the previous evaluations.

Figure 5 Iso-Mach lines in each subdomain , Mach= 0.75, Attack= 2.0^o



4 Results and Discussion

The methods presented have been tested for the solution of nonlinear transonic flows in a nozzle with two or three subdomains of two different mesh sizes. Significant improvements of the implementation with niching have been achieved as compared with the method without niching. The numerical results show that the new method based on local niching strategy can accelerate the speed of convergence and has the potential to remove the dependence of mesh size without a preconditioner (see [PC96] and [PMC96] for details).

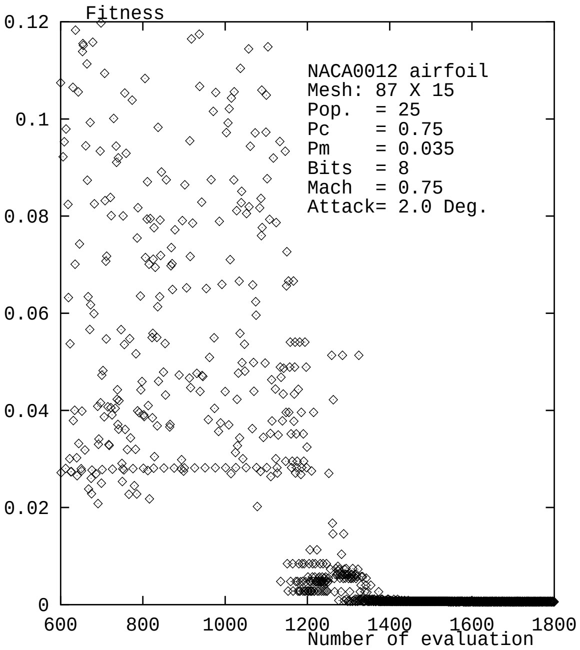
For the computations presented here, transonic flows past a lifting or non-lifting airfoil with multi subdomains are considered as further extension. The extension can be carried out in a straightforward way. The formulae of fitness function can be rewritten as follows:

$$J(\widetilde{g}_1, \widetilde{g}_2, ..., \widetilde{g}_k) = \sum_{i}^{K} J(\widetilde{g}_i) \qquad i \ = \ 1 \ , \ K \tag{4.5}$$

where K is the number of subdomain and $\widetilde{g}_i$ are the decoded values of genetic g_i representation on each interface of subdomains.

As shown in Figure 2, we decompose the whole domain in 5 subdomains and hence with 5 marked overlappings, which are placed symmetrically. The C-mesh, consisting of 87 × 15 was used. This means that on each interface of overlapping we have more than 15 parameters. Based on the local niching strategy, we choose one parameter for each interface as the key parameter. In the present case , 5 key parameters, which are located on the body surface, are to be coded to form a multi parameter string. It should be emphasized that two of these points are located just at the trailing edge, where the cutting lines are located in order to have unique potential. As a result, the circulation can be calculated based on potential values of these two coded points. This means that the lifting circulation is coded implicitly and is fixed for each evaluation, which is different from the traditional method for iteratively determining the circulation. Thus the present case is quite complicated as compared with that used in the nozzle.

Figure 6 lifting case , Mach= 0.75, Attack= 2.0^o



The numerical results for both non-lifting case with Mach= 0.80 and angle of attack = 0^o and lifting case with Mach= 0.75 and angle of attack = 2.0^o are presented in the Figures 3-6. It should be noted that the results of the non-lifting case are obtained by forcing symmetric conditions, such as zero circulation, which results in reducing the search space, thus it appears to have fast convergence as compared with that of lifting case. The iso-Mach lines of figures show the continuity of the solution in the overlapping subdomains. It should be mentioned that the method presented can benefit parallelism from both GAs and domain decomposition methods.

5 Conclusion

The conventional Schwarz iterative method can be extended to nonlinear situations with the genetic treatment on the interface of subdomain without preconditioner and the method presented has the potentiality to avoid the effect of the size of discretization. The solutions of other PDEs using the same concept of GAs are currently underinvestigation.

Acknowledgement

The authors would like to thank Professors, R. Glowinski, Y. Kuznetsov, and O. Pironneau, for the fruitful discussion about the robustness of GAs as preconditioner

companions of classical domain decomposition methods.

REFERENCES

[Gol89] Goldberg D. (1989) *Genetic Algorithms in Search Optimization and Machine Learning.* Addison-Wesley, Reading, Mass.

[Hol92] Holland J. (1992) *Adaptation in natural and artificial systems.* MIT Press.

[PC96] Periaux J. and Chen H. Q. (1996) Domain decomposition method using genetic algorithms for solving transonic aerodynamic problems. In Glowinski R., Périaux J., Shi Z., and Widlund O. (eds) *Proc. Eighth Int. Conf. on Domain Decomposition Decomposition Meths.* Wiley and Sons, Chichester.

[PMC96] Periaux J., Mantel B., and Chen H. Q. (1996) Genetic algorithms applied to domain decomposed flow computations. In Désidéri R.-A., Hirsch C., Tallec P., Pandolfi M., and J.-Périaux (eds) *Computational Fluid Dynamics '96 Proc. Third ECCOMAS Computational Fluid Dynamics Conf.* Wiley and Sons, Paris.

The Overlapping Component Mode Synthesis Method: The Shifted Eigenmodes Strategy and the Case of Selfadjoint Operators with Discontinuous Coefficients

Isabelle Charpentier, Florian De Vuyst, and Yvon Maday

1 Introduction

The Component Mode Synthesis (CMS) method is a domain decomposition strategy for the approximation of eigenmodes of partial differential elliptic operators. It makes use of local functions that are the eigenmodes of the same global operator but restricted over each subdomain. The first local eigenfunctions suitably extended over the whole domain (plus eventually some interface modes, see [CB68] for more details) are then used to span a discrete space that allows one to approximate the global eigenmodes through a Galerkin-type strategy. Whereas the standard method, based on a nonoverlapping domain decomposition, is of low order of accuracy, our variant [CDM96a] relies on an overlapping domain decomposition

$$\Omega = \bigcup_{k=1}^{K} \Omega^k, \tag{1.1}$$

and produces a method of infinite order accuracy (see below for the definition). The analysis and the first results we have presented in [CDM96a] deal with the case of a constant coefficient operator and the computation of the spectrum, starting from the lowest eigenmode. We generalize here the domain of application of the method by first introducing the shifted eigenmode strategy and secondly by considering elliptic operators with discontinuous coefficients.

This paper is divided as follows:

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

- In section 2 is discussed the shifted eigenmodes strategy. One and two-dimensional numerical tests are discussed.
- In section 3 are discussed in detail the additional problems arising with the presence of nonconstant coefficients.

2 Computing Eigenmodes of Energy Close to a Given One

For both engineering and mathematical purposes, it is sometimes attractive only to compute a part of the spectrum: low, medium or high frequencies. Conventional finite element or component mode synthesis methods are known not to be so efficient in the approximation of large eigenvalues. Indeed, in order to approximate high frequency eigenmodes, the trial functions of the discrete spaces have to be able to reproduce these modes. This generally requires high dimensional discrete spaces inducing a corresponding algebraic system that quickly becomes very large. Hence, the restriction in spectrum comes from the restriction in computational range.

Provided that all the first local eigenmodes are present in the discrete space, it has been shown in [CDM96a] that the overlapping method provides an infinite order of accuracy in the following sense: suppose that all the eigenmodes of energy less than $\widetilde{\lambda}$ are used; then the convergence is controlled by a constant times $(\frac{\overline{\lambda}}{\widetilde{\lambda}})^p$ for $\underline{\text{any}}$ p for the approximation of all the global eigenmodes of energy less than $\overline{\lambda}$.

We here prove that the use of the only local basis functions with energy close to the expected value is sufficient to capture the expected mode with an infinite order of accuracy.

For the sake of simplicity, we present this "*shifted eigenmodes strategy*" applied to the Laplace operator, but it can be extended to any linear elliptic selfadjoint operator.

Presentation and Numerical Analysis of the Shifted Eigenmodes Strategy

Let $\lambda^\star$ be a given positive real value. We are interested in the following problem: *find a pair* $(\lambda, u) \in R^+ \times H_0^1(\Omega)$ *such that* λ *is close to* $\lambda^\star$ *and*

$$\left\{ \begin{array}{rcl} -\Delta u & = & \lambda\ u \text{ in } \Omega, \\ u & = & 0 \text{ over } \partial\Omega. \end{array} \right. \tag{2.2}$$

On the subdomain Ω^k, we consider the same problem as (2.2) quoted here as $(2.2)^k$ where Ω is replaced by Ω^k. We denote by $\{\lambda_i^k, u_i^k\}_{i=1,\infty}^{k=1,K}$ the corresponding eigenmodes ranged in *increasing* order of eigenvalue. We denote by $\tilde{u}_i^k$ the extension of u_i^k by 0 over Ω.

Definition - Let α be a positive constant. The shifted mode strategy consists in considering a Galerkin method based on the discrete space $X_{\alpha,\lambda^\star}$ spanned by all local eigenmodes $\{\tilde{u}_i^k\}_{i=m_k,M_k}^{k=1,K}$, where m_k and M_k $(m_k < M_k)$ are some integers chosen such that

$$\lambda_{m_k}^k \ < \ \lambda^\star - \alpha \ < \ \lambda^\star + \alpha \ < \ \lambda_{M_k}^k \quad \text{for all integer } k \text{ in } \{1, ..., K\},$$

This problem is defined by: *find a pair* $(\mu, v) \in R^+ \times X_{\alpha,\lambda^\star}$ *such that*

$$\forall w \in X_{\alpha,\lambda^\star}, \int_\Omega \nabla v \nabla w = \mu \int_\Omega vw. \tag{2.3}$$

We now give the following result:

Proposition [error estimate] - *There exists a positive constant* $C(p)$, $p \in N$ *such that for any solution* (u, λ) *of (2.2) with*

$$\lambda^\star - \alpha \; < \; \lambda \; < \; \lambda^\star + \alpha,$$

the following error estimate holds for one of the eigenmode (μ, v) *of problem (2.3):*

$$\|u - v\|_{H_0^1(\Omega)} \leq \; C(p) \; \left[\left(\frac{\lambda_{m_k - 1}}{\lambda} \right)^p + \left(\frac{\lambda}{\lambda_{M_k+1}} \right)^p \right] \; \textit{for any } p \in N. \tag{2.4}$$

The above estimate expresses that the method is of infinite order of accuracy since it is better than any fixed order.

Sketch of the proof. - From the standard abstract results on the numerical analysis of the Galerkin approximation, it is well-known (see Châtelin [Cha83] for example) that the proof of the previous proposition reduces itself into the evaluation of the distance (in the H_0^1-norm)

$$\text{dist}\left(u, \; \text{span}(u_i^k)_{i=m_k, M_k}^{k=1,K} \right).$$

In order to evaluate this distance, we follow the same strategy as in [CDM96a]. A regular partition of unity $\{\chi_k\}_{k=1,K}$ is first associated to the domain decomposition of Ω. We are now looking for the ability of the functions $(u_i^k)_{i=m_k,M_k}$ to approximate the function $\chi_k u$. Because the set of functions $\{u_i^k\}_{i=1,+\infty}$ spans the space $H_0^1(\Omega^k)$, there exists a ℓ_2-summable family of coefficients $\{\alpha_i^k\}_i$ such that

$$\chi_k u \; = \; \sum_{i=1}^{+\infty} \alpha_i^k \; \tilde{u}_i^k.$$

We then approximate $(\chi_k u)$ by a truncated series

$$\chi_k u \; \approx \; \tilde{u}^k \; \stackrel{\text{def}}{=} \; \sum_{i=m_k}^{M_k} \alpha_i^k \; \tilde{u}_i^k,$$

so that the term $\sum_{k=1}^K \sum_{i=m_k}^{M_k} \alpha_i^k \, \tilde{u}_i^k$ will be a candidate for bounding the distance from above. We are left to estimate in the H_0^1-norm the quantities

$$\chi_k u \; - \; \tilde{u}^k \; = \; \left\{ \sum_{i=1}^{m_k - 1} \alpha_i^k \; \tilde{u}_i^k \; + \; \sum_{i=M_k+1}^{+\infty} \alpha_i^k \; \tilde{u}_i^k \right\}.$$

Two residual terms are present. The second one was already considered in the analysis of the initial version of the method (see [CDM96a]) and decays exponentially fast:

$$\left\| \sum_{i=M_k+1}^{+\infty} \alpha_i^k \ \tilde{u}_i^k \right\|_{L^2(\Omega)} \le c(p) \left(\frac{\lambda}{\lambda_{M_k+1}} \right)^p \quad \forall p \in N. \tag{2.5}$$

Recalling that u is an eigenvalue of the original problem, we write

$$\begin{aligned} \alpha_i^k &= \int_{\Omega^k} (\chi_k u) \ u_i^k \\ &= \int_{\Omega} u \ (\chi_k \ \tilde{u}_i^k) \\ &= \frac{1}{\lambda} \int_{\Omega} -\Delta u \ (\chi_k \ \tilde{u}_i^k) \\ &= \frac{1}{\lambda} \int_{\Omega} u \left[-\Delta(\chi_k \ \tilde{u}_i^k) \right]. \end{aligned}$$

Iterating this argument p times leads to

$$\alpha_i^k = \frac{1}{\lambda^p} \int_{\Omega} u \ [-\Delta]^p \, (\chi_k \ \tilde{u}_i^k) \tag{2.6}$$

From $(2.2)^k$, there exists a positive constant $C(p)$ such that

$$\| \, [-\Delta]^p \, (\chi_k \ u_i^k) \|_{L^2(\Omega)} \le C(p) \ \left(\lambda_{m_k-1} \right)^p, \tag{2.7}$$

and the result thus follows from (2.5), (2.6) and (2.7).

One-dimensional Numerical Tests

In this section we consider the following eigenvalue problem with constant coefficients: *find all pairs* $(\lambda, u) \in R^+ \times H_0^1(]0,1[)$ *such that*

$$\begin{cases} -\, u''(x) \ = \ \lambda \ u(x) \ \ \textit{for all } x \textit{ in }]0,1[, \\ u(0) = u(1) = 0. \end{cases}$$

Let us assume that $]0,1[$ is split into K overlapping sets

$$]0,1[\, = \bigcup_{k=1}^{K} \,]a_k, b_k[.$$

Here, the exact local eigenvalues are known: $\lambda_i^k = \left[\frac{i\pi}{b_k - a_k} \right]^2$. This will help us in the numerical experiments for verifying the infinite order of convergence. On each subdomain $\Omega^k =]a_k, b_k[$, we denote by N_k the number of consecutive local eigenmodes between λ_{m_k} and λ_{M_k} of the partial differential problem set on Ω^k, surrounding the eigenvalue $\lambda^\star$ we want to identify.

Table 1 Identification of the spectrum $\{\lambda_{min}, .., \lambda_{max}\}$ with precision of order 10^{-6} for decompositions D_1 and D_2, and with precision 10^{-4} for D_3.

	D_1	D_2	D_3
N	60	52	51
$\lambda^\star$=100	$\{88, .., 120\}$	$\{86, .., 121\}$	$\{91, .., 115\}$
$\lambda^\star$=200	$\{185, .., 220\}$	$\{183, .., 221\}$	$\{189, .., 221\}$
$\lambda^\star$=400	$\{385, .., 420\}$	$\{383, .., 421\}$	$\{386, .., 421\}$
$\lambda^\star$=510	$\{495, .., 531\}$	$\{493, .., 531\}$	$\{496, .., 531\}$

¿From the previous numerical analysis (see (2.4)), it is natural to tune the different pairs $(N_k, \lambda_{m_k})_{k=1,..,K}$ such that

$$\frac{N_1}{\mid \Omega^1 \mid} \approx \frac{N_2}{\mid \Omega^2 \mid} \approx \cdots \frac{N_K}{\mid \Omega^K \mid}$$

and

$$\frac{\lambda_{m_k}}{\lambda^\star} \approx \frac{\lambda^\star}{\lambda_{M_k}} \text{ for } k = 1, \dots, K.$$

It thus appears that the accuracy of the discretization only depends on the parameter $n = \frac{N_1}{|\Omega^1|}$ since all the others are then deduced from the relations

$$\frac{N_k}{\mid \Omega^k \mid} = n, \quad \frac{\lambda_{m_k}}{\lambda^\star} = \frac{\lambda^\star}{\lambda_{m_k} + N_k \, \pi^2} \quad \text{for} \quad k = 1, .., K. \tag{2.8}$$

Numerical experiments - We use our overlapping CMS method on the three following interval decompositions of Ω to highlight its accuracy:

$$\begin{cases} D_1 & : \emptyset =]0, 0.4[\cup]0.3, 0.7[\cup]0.6, 1[\,, \quad K = 3, \\ D_2 & : \emptyset =]0, 0.75[\cup]0.5, 1[\,, \quad K = 2, \\ D_3 & : \emptyset =]0, 0.6[\cup]0.6, 1[\cup]0.3, 0.7[\,, \quad K = 3. \end{cases}$$

Table 1 indicates the part of the spectrum surrounding the eigenvalue $\lambda^\star$ that has been identified with a relative error of order 10^{-6} or 10^{-4}.

Remark [unexpected discrete eigenvalues] - Particular unexpected eigenvalues may appear in the approximate spectrum. The abstract results on the approximation of eigenmodes (see Châtelin [Cha83]) indicate that (in all good cases) there exists a sequence of discrete eigenvalues that converges towards each exact eigenvalue. This does not prevent that, at some fixed discretization, there may exist spurious eigenmodes that eventually will converge towards an exact one that may be still far away. The problem is more present in the Shifted method than in the non shifted one since the convergence is not monotonic. We are indeed not certain that the discrete eigenvalues are larger than the exact ones.

In order to cure the problem, we propose two different solutions.

1. Let A be the (discrete) stiffness matrix built from *all* the first local eigenmodes. For each approximate eigenpair $(\tilde{\lambda}, \tilde{u})$, we compare the vector $(A\,\tilde{u} - \tilde{\lambda}\,\tilde{u})$ to zero (using the euclidian ℓ_2-norm for example); if the difference is small, then $(\tilde{\lambda}, \tilde{\mu})$ is a good approximation of an exact eigensolution. But this technique is expensive since it requires to build up the complete (large) matrix that we expected to avoid. Nevertheless, it is not so expensive as computing the eigenmodes of the total matrix A.

2. Another cheaper strategy is to define a set of test cases with slight variations of parameters (number of local basis functions, size of subdomains) and compare the different results. It is reasonable to think that an unexpected approximate eigenvalue is strongly dependent on the parameters of a computation and then will appear or disappear for slight variations of the parameters. This is a manner to localize these spurious modes. We have experimented this approach and it has given good results.

The Shifted Eigenmodes Strategy for Two-dimensional Problems

We extend here the shifted eigenmodes strategy to the multidimensional case. As before, it consists in choosing appropriate local modes according to the eigenvalue we want to approximate.

For the numerical experiments, we present the method using the Laplace operator defined on the unit square. The eigenvalues are analytically known:

$$\lambda_{kl} = (k^2 + \ell^2)\,\pi^2, \quad k, \ell \in N.$$

The unit square is splitted up into three overlapping subdomains

$$\cup_{k=1}^{3} \Omega^k \; =]0, 0.6[\times]0, 1[\;\cup\;]0.4, 1[\times]0, 0.7[\;\cup\;]0.25, 1[\times]0.4, 1[.$$

We denote by λ_{m_k} and λ_{M_k} two eigenvalues of the problem

$$-\Delta u \;=\; \lambda\, u \text{ on domain } \Omega^k, \tag{2.9}$$

$$u_{|\partial\Omega^k} \;=\; 0, \tag{2.10}$$

such that, as suggested by (2.4),

$$\frac{\lambda_{m_k}}{\lambda^\star} \simeq \frac{\lambda^\star}{\lambda_{M_k}}, \tag{2.11}$$

where $\lambda^\star$ is the eigenvalue we are interested in. Since the closed form of the global solution is known, the relation between the number of local modes N_k and the other parameters is roughly

$$N_k \;\approx\; \frac{\lambda_{M_k} - \lambda_{m_k}}{\mid \Omega^k \mid}. \tag{2.12}$$

As in the one-dimensional case, a coherent choice of the number of local eigenvalues determines the accuracy of the CMS method.

In order to identify a significant part of the spectrum around the global eigenvalue $\lambda^\star = 600\,\pi^2$, we first select 331 local modes of problem $(2.2)^1$ in Ω^1 dispatched around the local eigenvalue $\lambda^\star$ with respect to the relations (2.11), (2.12): those give

the number of modes on each other subdomain: 231 on Ω^2 and 247 on Ω^3. With these 331+231+247=809 local modes we compute the eigenvalues around $\lambda^\star$ using the shifted CMS method. A similar process is used to select a local basis made of 1355 local modes for the identification of the global eigenvalue surrounding $1000\,\pi^2$.
The following Figures (1 and 2) represent all the couples $(k, \ell) \in N^2$ that define the exact eigenvalues of the Laplace operator for the problem set in the unit square. Each square point (k, ℓ) indicates that there exists a computed eigenvalue close to the corresponding exact eigenvalue $((k^2 + \ell^2)\pi^2)$: a square point is plotted as soon as the relative error on the eigenvalue is less or equal than 2.10^{-3}. The black squared points show the couples (k, ℓ) such that $(k^2 + \ell^2)$ is close to 600, 1000 or 1500. We have indicated in Figure 4 the accuracy of each identified eigenvalue.

Figure 1 Identified indices (k, l) of eigenvalue $\lambda_{kl} = (k^2 + \ell^2)\,\pi^2$ around $600\,\pi^2$

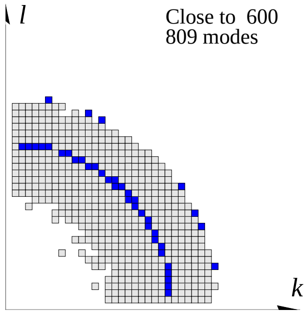
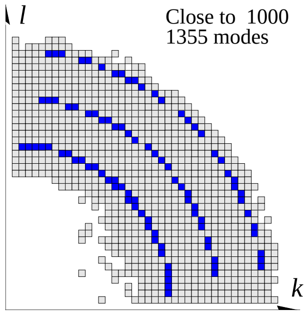


Figure 2 Identified indices (k, l) of eigenvalue $\lambda_{kl} = (k^2 + \ell^2)\,\pi^2$ around $1000\,\pi^2$



For five global eigenvalues around $\lambda^\star = 600\,\pi^2$, Figure 5 shows the accuracy of the computation. It is drawn the logarithm of the error between the exact eigenvalue and its approximate computed value with respect to the total number of local modes. This confirms the exponential convergence of the method.

Remark [parallel implementation] - In order to identify a large number of eigenvalues, there exist two solutions. The usual one consists in choosing larger local basis sets. This rapidly becomes too onerous in terms of memory and CPU time.
The second solution, suggested by the shifted eigenmodes strategy, consists in splitting up the expected part of the spectrum into several overlapping frequency bands. We then carry out as many independent calculations as there exists some bands. On

Figure 3 Identified eigenvalues (k, l) (no shift)

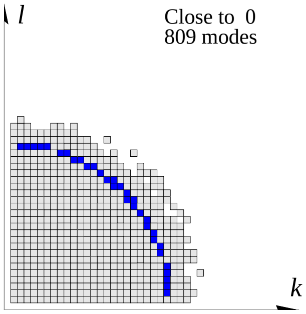


Figure 4 Accuracy of the method

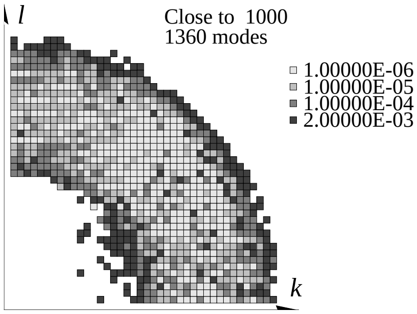
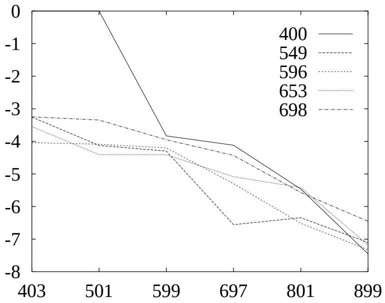


Figure 5 Accuracy of the method



each band, we apply the shifted eigenmodes strategy. Now, the involved matrices are of smaller size and the global algorithm complexity is less than the one of the computation of the whole large band.

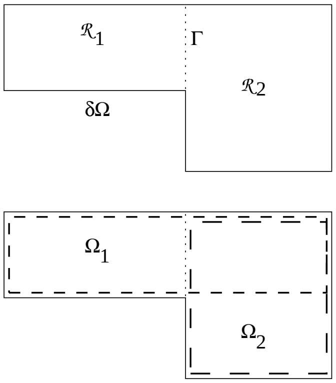
As an illustration, we have considered the test using 809 unshifted local modes. It appears from the Figures 3, 1 and 2 that this parallel approach is a viable tool for computing the whole spectrum from the smallest eigenvalue up to those close to $1500\,\pi^2$.
It is then quite easy to consider parallel algorithms to carry out computations that are completely independent.

Conclusion - The Shifted Component Modes Synthesis (SCMS) method allows for computing eigenmodes corresponding to large eigenvalues without requiring the approximation of all the smaller eigenvalues and without loosing the infinite order of accuracy. It also provides features adapted to the parallelization of the computation of the spectrum. It is interesting to note that the problem of the mass matrix bad conditioning already reported in previous papers (see [CDM96b],[CDM96a]) is weakened by the use of this technique.

3 The Overlapping CMS Method for Operations with Discontinuous Coefficients

As it is often the case in industrial problems, the related partial differential equations are generally set on a quite complex domain and the coefficients involved in the operator are often nonconstant. They may even present some discontinuities. Let us consider here for example the problem of the vibration of a membrane made of two rectangular membranes $\mathcal{R}_1$ and $\mathcal{R}_2$ as it is drawn below. Each rectangular membrane $(\mathcal{R}_i)$, $i = 1, 2$, has its own characteristic of vibration expressed in terms of a constant a_i, $i = 1, 2$. We denote by Γ the boundary between $\mathcal{R}_1$ and $\mathcal{R}_2$. The membrane is fixed at the domain boundary $\partial\Omega$. In order to approximate the eigenmodes on this

Figure 6 Overlapping domain decomposition on a "L-shaped" structure involving two materials.



membrane, we decide to use a strategy of subdomains decomposition with overlapping (Ω_1, Ω_2) as proposed in [CDM96a]. The overlapping strategy avoids the problem of the definition of local interface basis functions. Otherwise, the decomposition leads to an operator T_k, $k = 1, 2$ defined on Ω^k with discontinuous coefficients at the interface Γ. As before, the analysis of the convergence rate can be achieved.

To solve the problem, one can use an overlapping subdomain with either constant or nonconstant coefficients. The convergence of the CMS method is thus of finite order of accuracy for the first choice and of infinite order for the second one. Of course, in the present situation, a camembert-shaped domain has to be added as suggested in [CDM96a] to solve the corner singularity.

Setting of the problem - Let $\Omega =]0, x_a[\times]0, y_b[$, $x_a, y_b > 0$. Let α be a constant, $0 < \alpha < 1$. We are interested in the solutions of the following eigenvalue problem: *find a pair* $(\lambda, u) \in R^+ \times H_0^1(\Omega)$ *such that*

$$\left\{ \begin{array}{rcl} -\operatorname{div}(A\,\nabla u) & = & \lambda\, u \text{ in } \Omega, \\ u_{|\partial\Omega} & = & 0, \end{array} \right. \tag{3.13}$$

where $A(x, y) = a_1 > 0$ if $x > \alpha\, x_a$, $A(x, y) = a_2 > 0$ otherwise. For the sake of simplicity, we set $\mathcal{R}_1 =]0, \alpha\, x_a[\times]0, y_b[$ and $\mathcal{R}_2 =]\alpha\, x_a, x_a[\times]0, y_b[$. The solutions cannot be explicitly written in a fully closed form, but they can be approximated as close as we want to the exact solutions. More precisely, we have proved :

Proposition 1 [eigenvalues] - *There exists an infinite countable set of eigenvalues of problem (3.13). These eigenvalues* $\{\lambda_{kl}\}_{l=1,+\infty}$ *(k is a positive integer) are characterized by the relations*

1. *If* $\lambda_{kl} \geq \max(a_1\,(k/y)^2\,\pi^2,\; a_2\,(k/y)^2\,\pi^2)$ *, then* λ_{kl} *are solutions of the equation*

$$a_1\; g_1(\lambda_{kl}) \tan\left(g_2(\lambda_{kl})\,(1-\alpha)\,x_a\right) \;+\; a_2\; g_2(\lambda_{kl}) \tan\left(g_1(\lambda_{kl})\,\alpha\,x_a\right) \;=\; 0, \tag{3.14}$$

with $g_i(\lambda) \overset{def}{=} \sqrt{\frac{\lambda - a_i\,(k/y_b)^2\,\pi^2}{a_i}}$*,* $i = 1, 2$*.*

2. *If* λ_{kl} *is such that* $\min(a_1\,(k/y)^2\,\pi^2,\; a_2\,(k/y)^2\,\pi^2) \leq \lambda_{kl} \leq \max(a_1\,(k/y)^2\,\pi^2,\; a_2\,(k/y)^2\,\pi^2)$*, then it is solution of the equation*

$$a_1\; h_1(\lambda_{kl}) \tanh\left(h_2(\lambda_{kl})\,(1-\alpha)\,x_a\right) \;+\; a_2\; h_2(\lambda_{kl}) \tan\left(h_1(\lambda_{kl})\,\alpha\,x_a\right) \;=\; 0, \tag{3.15}$$

where $h_i(\lambda) \overset{def}{=} \sqrt{\frac{|\lambda - a_i\,(k/y_b)^2\,\pi^2|}{a_i}}$*,* $i = 1, 2$*.*

When the determination of each eigenvalue is achieved, through the solution of equation (3.14) or (3.15) , the eigenmodes can be obtained in closed form. It is easy to prove that the solutions restricted to each subdomain $\mathcal{R}_1$ or $\mathcal{R}_2$ are either sin or sinh functions. The constants of integration are determined via the boundary conditions and continuity conditions for both u_{kl} and $A(.)\nabla u_{kl}$ at $x = \alpha\, x_a$:

Proposition 2 [eigenmodes] - *The eigenvalues* u_{kl} *of problem (3.13) are given by*

1. *If* $\lambda_{kl} \geq \max(a_1\,(k/y)^2\,\pi^2,\, a_2\,(k/y)^2\,\pi^2)$*, then*

$$\left\{\begin{array}{lcl} u_{kl}(x,y) &=& B_{12}(\lambda)\ \sin(g_1(\lambda_{kl})\,x)\sin(k\,\pi\,y) \\ && \qquad\qquad \textit{if } x \leq\ \alpha x_a, \\ u_{kl}(x,y) &=& \sin(g_2(\lambda_{kl})\,(x_a - x))\sin(k\,\pi\,y) \\ && \qquad\qquad \textit{otherwise,} \end{array}\right. \tag{3.16}$$

with $B_{12}(\lambda) = \sin\left(g_2(\lambda)\,x_a\,(1-\alpha)\right) / \sin\left(g_1(\lambda)\,x_a\,\alpha\right)$.

2. *If* $\min(a_1\,(k/y)^2\,\pi^2,\, a_2\,(k/y)^2\,\pi^2) \ \leq\ \lambda_{kl}\ \leq\ \max(a_1\,(k/y)^2\,\pi^2,\, a_2\,(k/y)^2\,\pi^2)$*, then (in case* $a_1 < a_2$*)*

$$\left\{\begin{array}{l} u_{kl}(x,y) = \sin(h_1(\lambda_{kl})\,x)\sin(k\,\pi\,y) \\ \qquad\qquad\qquad\qquad \textit{if } x \leq\ \alpha x_a, \\ u_{kl}(x,y) = B_{21}(\lambda)\ \sinh(h_2(\lambda_{kl})(x_a - x))\ \sin(k\,\pi\,y) \\ \qquad\qquad\qquad\qquad \textit{otherwise,} \end{array}\right. \tag{3.17}$$

with $B_{21}(\lambda) = \sin\left(h_1(\lambda)\,x_a\,\alpha\right) / \sinh\left(h_2(\lambda)\,x_a\,(1-\alpha)\right)$*. (the case where* $a_1 > a_2$ *is completely symmetric).*

Remark [lack of regularity of the solutions] - The closed forms of the eigenmodes on the rectangular domain show that the solutions are not generally smooth. Indeed, whenever $a_1 \neq a_2$, the normal derivative of $A(.)\nabla u_{kl}$ is continuous at the interface $x = \alpha$, but it is not generally the case for the function ∇u_{kl}. Thus $u_{kl} \notin \mathrm{H}^2(\Omega)$. The singularities are then localized on the lines of discontinuity of the function $A(.)$. This limitation of regularity of solutions would drastically limit the order of accuracy of a standard method of approximation. We have considered the following test: $a_1 = 1$, $a_2 = 5$, $\alpha = 1/2$, $x_a = 1/2$ and $y_b = 1$.

A dichotomy algorithm has been implemented in order to find the roots of the two equations (3.14),(3.15) up to the precision of the computer.

On table 2 we display the 30 first computed eigenvalues (arranged in increasing order). One can notice that, for the modes number 7 and 29, the eigenvalues are exactly $40\,\pi^2$ and $160\,\pi^2$ respectively. For example, mode 7 is the function $u_7(x,y) = \sin(6\pi x)\cos(2\pi y)\mathbb{1}_{0\leq x\leq 1/4} + \sin(2\pi x)\cos(2\pi y)\mathbb{1}_{1/4\leq x\leq 1/2}$. At the interface $x = 1/4$, both functions u_7 and ∇u_7 are continuous (since ∇u_7 vanishes) hence all the successive derivatives of u_7 are continuous too.

Let us also observe that some eigenvalues can be very close together. It is the case for modes number 19 and 20, and also 29 and 30 (see table 2). It is interesting to check that the method correctly captures these approximate eigenvalues and eigenmodes. We plot below the eigenmodes associated to eigenvalues number 19 and 20. The corresponding modes are different.

4 Numerical Results for Operators with Discontinuous Coefficients

Our purpose is now to illustrate numerically the accuracy of different CMS methods in case of discontinuous coefficients. On the unit square domain Ω, we decide to

Table 2 Exact eigenvalue for Test Case 1 (up to the computer precision)

eig. nb	eigenvalue / π^2	eig. nb	eigenvalue / π^2
1	11.381858321428	16	86.561289001402
2	15.874044219346	17	93.126493282476
3	22.045698340432	18	96.014731853507
4	29.771455839467	19	107.36015972229
5	30.811703131273	20	107.65137053876
6	39.222849777720	21	115.11237794574
7	40.0	22	123.19446535873
8	50.522457182429	23	131.67906445402
9	53.599749966838	24	134.06922677463
10	63.734486614315	25	137.31868568331
11	67.387028033897	26	140.80059050878
12	68.390060393702	27	146.50622200202
13	74.340847129365	28	157.47087545240
14	78.892410930881	29	160.0
15	80.106572892321	30	160.26355418007

Figure 7 Exact mode number 19 of the test case. $\lambda_{19} = 107.36$

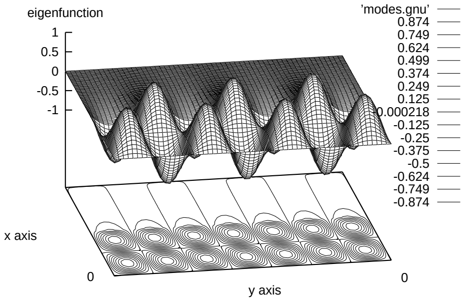


Figure 8 Exact mode number 20 of the test case. $\lambda_{20} = 107.65$

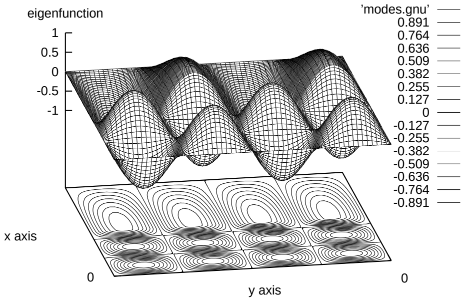
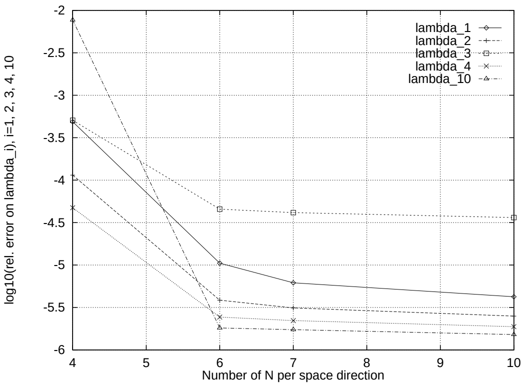


Figure 9 Relative errors on different eigenvalues for different numbers of local modes on the overlapping subdomain



approximate the first eigenvalues of problem (3.13). As in the previous example, we consider a domain decomposition using two nonoverlapping rectangular subdomains with interface localized along the discontinuity line of coefficients and a third overlapping subdomain that covers the interface. We choose

$$\Omega_1 = (0, 0.5) \times (0, 1) \quad (a_1 = 5), \tag{4.18}$$

$$\Omega_2 = (0.5, 1) \times (0, 1) \quad (a_2 = 1), \tag{4.19}$$

$$\Omega_3 = (0.25, 0.75) \times (0, 1) \quad \text{(overlapping subdomain)}. \tag{4.20}$$

For the numerical experiments, we consider:

1. an overlapping subdomain subdomain with a constant coefficient $a(x, y) = a_3 = 1$;
2. an overlapping subdomain with nonconstant coefficients:

$$a(x, y) = 5 \;\; \forall (x, y) \in (0.25, 0.5), \tag{4.21}$$

$$a(x, y) = 1 \;\; \forall (x, y) \in (0.5, 0.75). \tag{4.22}$$

On each subdomain Ω_i $(i = 1, 2)$, we consider 144 local modes that correspond to

$$\sin(2k\pi x)\sin(l\pi y) \quad for \ (k, l) \in \{1, .., 12\}^2.$$

For the first test, we consider an overlapping subdomain with constant coefficient. In Table (3) we give the computed eigenvalues with this approach and we plot the relative errors computed with respect to the accurate eigenvalues obtained by dichotomy (see below) in Figure 9. We immediately observe that for the overlapping domain with constant coefficient, the relative errors are bounded by 10^{-4}. But the curves ("constant 6 × 6", "constant 10 × 10") are quite similar. That confirms that the

Table 3 Approximate eigenvalues for Test Case 1

eig. nb	eigenvalue / π^2	eig. nb	eigenvalue / π^2
1	11.3819115	13	74.3415565
2	15.8741089	14	78.8925446
3	22.0457681	15	80.1086039
4	29.7715310	16	86.5643435
5	30.8124850	17	93.1282303
6	39.2229341	18	96.0148882
7	40.0015024	19	107.361750
8	50.5225546	20	107.661054
9	53.6021736	21	115.112560
10	63.7346006	22	123.196000
11	67.3894861	23	131.683773
12	68.3902544	24	134.086982

use of overlapping subdomain with constant coefficients has induced a finite order of convergence for the CMS method. In order to raise the rate of convergence, we naturally propose to use an overlapping subdomain with nonconstant coefficients. So in a first step we compute by the dichotomy method the first local modes (of Ω_3) made of sine functions and hyperbolic sine functions.

The preliminary numerical results are in good agreement with the improved accuracy of the method and we shall report in a future work the results of the numerical simulation for this problem.

In this future work, we shall also deal with another important example concerning the capture of singularities not only due to the presence of discontinuous coefficients, but also due to corners at the boundaries. Two ingredients have to be used then: the non constant coefficients on the overlapping domain and the presence of "camembert shaped" subdomains surrounding the corner singularities as it has been explained in [CDM96a].

REFERENCES

[CB68] Craig R. and Bampton M. (1968) Coupling of substructures for dynamic analysis. *A.I.A.A. Journal* 6: 1313–1321.

[CDM96a] Charpentier I., De Vuyst F., and Maday Y. (1996) A component mode synthesis method of infinite order of accuracy using subdomain overlapping : numerical analysis and experiments. Publication du laboratoire d'Analyse Numérique R96002, Université Pierre et Marie Curie, Laboratoire d'Analyse Numérique, Tour 55-65 5è étage, 4, place Jussieu, 75252 PARIS Cedex 05, FRANCE.

[CDM96b] Charpentier I., De Vuyst F., and Maday Y. (1996) Méthode de synthèse modale avec une décomposition de domaine par recouvrement. *Comptes Rendus de l'Académie des Sciences Série I* (t. 322): 881–888.

[Cha83] Chatelin F. (1983) *Spectral approximation of linear operators.* Academic Press, New York.

Robust Additive Schwarz Methods on Unstructured Grids

Petter E. Bjørstad, Maksymilian Dryja, and Eero Vainikko

1 Introduction

The purpose of this paper is to describe a fully parallel two and three-dimensional computer implementation of the Additive Average Schwarz algorithm first described in [BDV97]. This paper draws heavily on the Ph.D. thesis work of Eero Vainikko [Vai97] and his extensive work required to design and develop a general computer implementation with direct coupling to state of art graph partitioning software such as Chaco [HL95] and MeTiS [KK95].

Our algorithm and its implementation share several interesting properties not always available in more standard domain decomposition algorithms:

- It handles unstructured grids in two and three dimensions
- It allows subdomains to be completely unstructured
- It is fully parallel and portable
- It is robust with respect to discontinuous coefficients across subdomain boundaries
- It can take advantage of inexact solvers.

There has been a strong trend within the applied computational sciences [VSB91] and in the scientific computing community to consider the use of unstructured grids and discretizations [CSZ96]. This approach often offers considerable advantages, in particular, when attacking three-dimensional problems where adaptivity and accuracy requirements may change across the computational domain.

The flexibility offered by unstructured discretizations may come with a cost in terms of more expensive computational procedures and possibly less efficient parallel implementations. Unstructured meshes have been standard in structures calculations based on finite elements since its pioneering start about forty years ago, but always dependent on direct methods of solution. Recently, the study of iterative methods for structural analysis has received more attention partly motivated by the very large problems generated by detailed three-dimensional models.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

The most natural approach to domain decomposition algorithms for unstructured problems, the iterative substructure or Schur complement methods [SBG96], can be viewed as a direct line of development from the multilevel superelement technique used to organize a direct factorization method [Prz85]. In both cases the solution procedure reduces the problem to the interfaces of substructures by considering Schur complement matrices. The explicit formation of these matrices is generally avoided when using the iterative substructure approach. These methods belong to the nonoverlapping class of domain decomposition algorithms. Early work with the Neumann-Dirichlet algorithm can be found in [BW86] and more recent, very promising work related to the Neumann-Neumann algorithm can be found in [LT94], [Man93] and [TV97].

In comparison, relatively little has been reported on the use of overlapping Schwarz methods for unstructured grids. In [CSZ96], the authors use a standard (overlapping) Schwarz method and a sequence of non-matching coarser grids. In particular, a regular grid is used at the coarsest level. This choice is at least partly dictated by the standard theory for Schwarz methods which assumes that the coarse grid can be viewed as a finite element triangulation of the domain.

Our proposed algorithm, the Additive Average Method [BDV97] is an additive Schwarz method with an alternative coarse space. The method avoids overlapping subdomains and solves a special linear system on the interface variables. In this respect, the method resembles the iterative substructuring methods. Our algorithm computes average values on each subdomain. The use of average values in order to capture the coarse grid behavior and achieve almost optimal preconditioners for iterative substructuring algorithms was used in the important paper [BPS87] already ten years ago.

This paper is organized as follows. In Section 2, we quickly review the method and its basic properties. The reader should consult [BDV97] to see the proof of convergence. In Section 3, we discuss a computer implementation and show how our algorithm can be coupled to state of art graph partitioning software packages. We do not discuss computational complexity, but refer to [BDV96] where one can also find details on parallel computations with reported performance. In the last section, we report on realistic computational problems with unstructured grids. We investigate subdomain partitioning guided by material properties and report on numerical experiments carried out on different parallel machines.

2 The Method and Some Properties

Let us consider a polyhedral region $\Omega \subset R^d, d = 2, 3$, which has been divided into N subdomains. The subdivision may be based on the information about a coefficient $\rho_i > 0$ which is assumed to be constant in each subdomain but may have big jumps across subdomain boundaries. We consider an elliptic problem in the variational form:

Find $u^* \in V(\Omega)$ such that

$$\sum_{i=1}^{N} \int_{\Omega_i} \rho_i \nabla u^* \cdot \nabla v dx = \int_{\Omega} f v dx \quad \forall v \in V(\Omega), \tag{2.1}$$

where $V(\Omega)$ is an appropriate Sobolev space.

We assume that the region Ω is triangulated into elements $e_k,\ k = 1, 2, ..., n_{t.}$ Let V^h be a finite element space of piecewise linear, continuous functions defined on the triangulation and vanishing on $\partial\Omega$, the boundary of Ω . Our discrete problem reads:

Find $u \in V^h(\Omega)$ such that

$$a(u, v) = f(v) \quad \forall v \in V^h(\Omega), \tag{2.2}$$

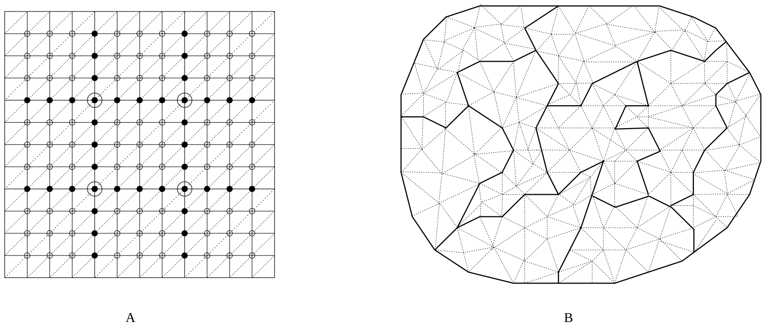
where

$$a(u, v) = \sum_{i=1}^{N} \int_{\Omega_i} \rho_i \nabla u \cdot \nabla v dx, \qquad f(v) = \int_{\Omega} f v dx. \tag{2.3}$$

In this paper we focus on the implementation and early experience with this method applied to unstructured grid problems.

The domain is divided into nonoverlapping subdomains. Figure 1 shows some key differences between a structured and unstructured grid in a domain decomposition context. On a regular grid it is straightforward to define a coarser space However, there are situations where for some essential reason, like *e.g.,* geological structure or material properties there is no natural regular splitting, in fact, one may want to decompose the domain according to problem specific considerations. With our method, we are free to do this as our coarse space construction requires no regular subdomain structure.

Figure 1 An example of a regular and an irregular splitting of a domain.



We partition the space V^h into $N + 1$ subspaces

$$V^h = V_0 + V_1 + ... + V_N$$

where $V^i = H_0^1(\Omega_i) \bigcap V^h$ and zero outside of Ω_i $(i = 1, ..., N)$. We denote the nodal points of $\partial\Omega_i$ and Ω_i by $\partial\Omega_{ih}$ and Ω_{ih}, respectively. Let n_i denote the number of nodes on $\partial\Omega_{ih}$. We define the coarse space V_0 by $V_0 = Range(I_A)$, the range of an interpolation-like operator I_A. For $u \in V^h$ restricted to $\overline{\Omega}_i$ we define $I_A u$ as follows:

$$I_A u = \begin{cases} u(x) & x \in \partial\Omega_{ih} \\ \overline{u}_i & x \in \Omega_{ih} \end{cases} \tag{2.4}$$

where

$$\overline{u}_i = \frac{1}{n_i} \sum_{x \in \partial\Omega_{ih}} u(x), \tag{2.5}$$

the average of the nodal values of u on the boundary of Ω_i.

We define by h_i the diameter of the smallest element touching the boundary $\partial\Omega_i$:

$$h_i = \inf_{j: e_j \cap \partial\Omega_i} h_j. \tag{2.6}$$

For each $V_i, i = 1, ..., N$ we introduce a bilinear form $b_i(u, v)$ on $V_i \times V_i$ of the form

$$b_i(u, v) = a(u, v). \tag{2.7}$$

An approximate bilinear form on the coarse space can be defined by

$$b_0(u, v) = \sum_i \rho_i h_i^{d-2} \sum_{x \in \partial\Omega_{ih}} (u(x) - \overline{u}_i)(v(x) - \overline{v}_i). \tag{2.8}$$

Denote the matrix obtained form the bilinear form $a(u, v)$ in (2.3) by A and let A_i $i = 1, ..., N$ denote the matrices defined by (2.7) *i.e.,* the restriction of A to the nodes of Ω_i. Similarly, the coarse grid matrix A_0 is computed from (2.8). Together with A_0 we also define another variant of the coarse matrix defined by

$$A_e = \tilde{I}_A^T A \tilde{I}_A \tag{2.9}$$

where $\tilde{I}_A^T$ is a restriction operator defined as the transpose of $\tilde{I}_A$, the matrix representation of I_A in (2.4). The matrix A_e can therefore be explicitly formed using only the definition of I_A. A_0 is an approximation of A_e, having sparse structure and it is therefore easier to use in computations. We call the use of A_e the Galerkin coarse space approximation.

Note that in the Additive Average method all the subdomain solves and the coarse problem solution can be run in parallel. Alternatively, we can define multiplicative variants of the method just as in the standard Schwarz methods. A simple, symmetric variant consists of a coarse solver followed by the subdomain solutions and another coarse solver at the end of the cycle. Residual updates must be carried out between the steps as usual.

When the Additive Average method is combined with a Krylov subspace iterative method, one can prove that the condition number of the resulting iteration matrix is independent of jumps in the coefficient ρ_i across subdomain boundaries and that it depends linearly on the ratio H/h where H is the largest subdomain diameter while

h measures the smallest diameter of an inscribed circle in an element that touches a subdomain boundary. The reader should consult [BDV97] to see a proof of convergence. Furthermore, it can be shown that the condition number depends linearly on the aspect ratio of the subdomains. Thus, in the case of a very long and thin subdomain it would be advantageous to further decompose this into smaller subdomains having a better aspect ratio.

3 Computer Implementation

Implementation of the Average Additive method as a parallel algorithm has several advantages over the standard Schwarz methods with respect to simplicity (no overlap), reduced arithmetic, and reduced communication. These issues are more fully explained in [BDV96].

The parallel implementation follows an SPMD programming model, where we assign several special tasks including the coarse space solver to processor zero. This processor will also handle code parameters and the initial triangulation of the domain that is needed to create the finite element data structures. In the regular case, the splitting of the domain into subdomains is straightforward. In the case of a nonregular domain, processor zero will create a graph where each element e_k is a vertex with edges connecting it to all vertices that correspond to element neighbors. We have coupled our code directly to the Chaco-2.0 [HL95] and the MeTiS [KK95] packages for graph partitioning. Both packages can be used to split the graph (and therefore our domain) into a specified number of pieces. The packages allow us to specify weights on the edges which can be used to supply information about the coefficients ρ_i. Ideally, this could be used to enforce a subdomain splitting where we guarantee that the ρ_i remains (near) constant in each subdomain with all the jumps across subdomain boundaries. In practice, this is somewhat difficult to achieve and more experience with the coupling of domain decomposition software and graph partitioning packages is needed.

The graph partitioning process gives us a discrete function $P : \{e_k, k = 1, ..., n_t\} \rightarrow \{1, ..., N\}$ that determines for each vertex (element) which subdomain it belongs to. In order to be able to assemble the stiffness matrix in parallel, subdomain number i is assigned information about its triangles $\{e_k : Pe_k = i\}$ together with one additional layer of the triangles around partition i. For each subdomain, we then send this information to an available processor. Next, we can perform the assembly of the stiffness matrix A in parallel. Each piece of the matrix A is finally stored where it was assembled. Each processor has also been given information about possible other subdomains that may 'own' one of its nodes. With this information the nodes can be split into sets of interior and boundary nodes respectively, and the necessary data structures for nearest neighbor as well as coarse solver communication can be prepared.

4 Numerical Experiments with Unstructured Meshes

Our first example considers an unstructured grid around an airfoil, see Figure 2. We subdivide the element mesh using the graph partitioning packages described

Figure 2 An unstructured mesh around an airfoil. The paper considers a similar mesh and two further levels of refinement.

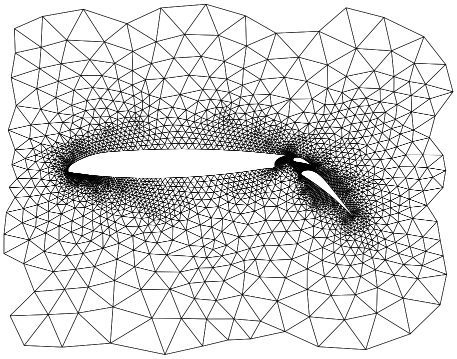


Table 1 Characteristics of the iterative method when applied to a scaled version of the airfoil mesh.

# PE	N	Dofs	$\kappa(BA)$	# Iter	Time/iter T3E	Time/iter Origin
4	3	15606	117	54	0.036	0.019
16	15	61484	636	119	0.030	0.026
28	27	244047	3000	252	***	0.059

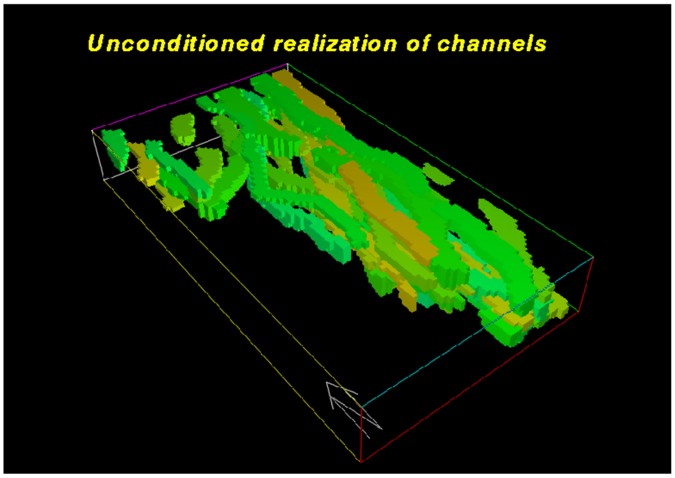
earlier. We further refine the mesh in order to scale the problem keeping roughly the same number of nodes per subdomain. In Table 1 we show iteration counts and condition number estimates for three different cases. We also show the execution time per iteration on a T3E and on an Origin-2000 computer when using only two symmetric Gauss-Seidel sweeps as our inexact solver. Observe that our code is completely unoptimized for these machines and that the execution times could change considerably in the near future. We note that the times are quite similar and that the time on the Origin scales correctly between the second and third row in the table. The T3E did not have enough memory per node to run our largest case.

We see that the condition number of our preconditioned problem does increase unlike the situation for a uniform refinement of a structured grid. A considerable part of this effect is due to our inexact solver, the entry in the second row of the table is reduced from 636 to 270 if we use an exact subdomain solver. This entry is further reduced to 165 by a simple diagonal scaling of the approximate bilinear form in (2.8). We also note that the quality of our graph partitioning (for example the subdomain aspect ratio) may change as we refine the problem and request more subdomains. The ratio H/h of our largest subdomain diameter relative to the smallest element is of the order 10^4 in this example, but tends to decrease as we refine our domain and introduce a larger number of subdomains. We plan to investigate the use of weights in the definition of our interpolation operator I_A as well as a more careful study of the quality of the mesh partitioning in order to further improve the situation.

Our last example is motivated by the oil industry. Figure 3 shows an oil reservoir model obtained from Norsk Hydro. The figure shows two distinctly different rock structures, with very different permeability. The geometry is quite irregular and complex in all three spacial dimensions. The entire domain is discretized into $55 \times 80 \times 100$ blocks and we use the permeability values to split the corresponding graph representation into two separate graphs which in turn are split into subdomains by our graph partitioning software. In this way, we obtain a highly irregular subdomain partitioning precisely tailored to our physical problem. We also split the problem in a completely regular fashion resulting in many subdomains having an internal discontinuous jump (from 1 to 1000) in the parameter ρ_i. In Table 2 we compare the results of four different experiments; our irregular subdomain partitioning combined with the Additive Average method, the same method using a regular splitting, a classical Additive Schwarz preconditioner as well as just running the conjugate gradient iteration without any preconditioning. In the three first cases we use two simple symmetric Gauss-Seidel sweeps as our inexact subdomain solver.

We first observe the large difference between the two first methods due to the internal jumps in the second method. The overlapping Schwarz method is considerably more robust, but still uses about 2.5 times as many iterations each of which is more costly than the iterations in the first method. Finally, we see that the problem requires a very large number of iterations without any preconditioning.

Figure 3 An oil reservoir model.



5 Conclusion

We have described a nonoverlapping additive Schwarz algorithm applied to unstructured grid problems. The method has interesting properties with respect to generality as well as efficient parallel implementations. Repeated refinements of an unstructured grid causes an increase in the condition number of our iteration operator. There may be many different factors contributing to this and we are currently trying

Table 2 Comparison of four different algorithms. The

Method	Decomposition	N	$\kappa(BA)$	# Iter
Additive Average	Irregular	510	694	117
Additive Average	Regular	512	169000	1610
Additive Schwarz	Regular	512	2330	292
No preconditioning	-	-	401000	2746

to identify these in order to further improve the algorithm.

We have demonstrated that an irregular domain decomposition adapted to the problem at hand can result in improved convergence of our iterative algorithm. Further work in progress includes scaling of the bilinear forms and the use of weights in the interpolation in order to improve the robustness when used on highly unstructured meshes. Our code is directly coupled to state of art graph partitioning packages. This interaction deserves more study in order to arrive at a best possible overall solution strategy.

REFERENCES

[BDV96] Bjørstad P. E., Dryja M., and Vainikko E. (December 1996) Parallel implementation of a schwarz domain decomposition algorithm. In Wasniewski J., Dongarra J., Madsen K., and Olesen D. (eds) *Applied Parallel Computing in Industrial Problems and Optimization*. Springer. Lecture Notes in Computer Science volume 1184.

[BDV97] Bjørstad P. E., Dryja M., and Vainikko E. (1997) Additive Schwarz methods without subdomain overlap and with new coarse spaces. In Glowinski R., Périaux J., Shi Z., and Widlund O. B. (eds) *Domain Decomposition Methods in Sciences and Engineering*. John Wiley & Sons. Proceedings from the Eight International Conference on Domain Decomposition Metods, May 1995, Beijing.

[BPS87] Bramble J. H., Pasciak J. E., and Schatz A. H. (1987) The construction of preconditioners for elliptic problems by substructuring, II. *Math. Comp.* 49: 1–16.

[BW86] Bjørstad P. E. and Widlund O. B. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal.* 23(6): 1093–1120.

[CSZ96] Chan T. F., Smith B. F., and Zou J. (1996) Overlapping Schwarz methods on unstructured meshes using non-matching coarse grids. *Numer. Math.* 73(2): 149–167.

[HL95] Hendrickson B. and Leland R. (July 1995) The Chaco user's guide. Version 2.0. Technical Report SAND94-2692, Sandia National Laboratories, Albuquerque, NM 87185-1110.

[KK95] Karypis G. and Kumar V. (August 1995) Metis, unstructured graph partitioning and sparse matrix ordering system. version 2.0. Technical report, University of Minnesota, Department of Computer Science, Minneapolis, MN 55455.

[LT94] Le Tallec P. (1994) Domain decomposition methods in computational mechanics. In Oden J. T. (ed) *Computational Mechanics Advances*, volume 1 (2), pages 121–220. North-Holland.

[Man93] Mandel J. (1993) Balancing domain decomposition. *Comm. Numer. Meth. Engrg.* 9: 233–241.

[Prz85] Przemieniecki J. S. (1985) *Theory of Matrix Structural Analysis*. Dover

Publications, Inc., New York. Reprint of McGraw Hill, 1968.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press.

[TV97] Tallec P. L. and Vidrascu M. (1997) Generalized Neumann-Neumann preconditioners for iterative substructuring. In Bjørstad P. E., Espedal M., and Keyes D. (eds) *Domain Decomposition Methods in Sciences and Engineering.* John Wiley & Sons. Proceedings from the Ninth International Conference, June 1996, Bergen, Norway.

[Vai97] Vainikko E. (March 1997) *Robust Additive Schwarz Methods – Parallel Implementations and Applications.* PhD thesis, University of Bergen.

[VSB91] Venkatakrishnan V., Simon H. D., and Barth T. J. (September 1991) A mimd implementation of a parallel euler solver for unstructured grids. Technical report, NASA Ames, Mail stop 202A NASA Ames Research Center, Moffett Field, CA 94305. RNR Technical Report RNR-91-024.

72

Overlapping Domain Decomposition with Non-matching Grids

Yuri A. Kuznetsov

1 Introduction

In this paper we consider two topics. In Section 2 we introduce a new macro-hybrid formulation based on overlapping domain decomposition for linear elliptic equations with symmetric positive definite operators. The problem is discretized by the mortar element method using non-matching grids on the interfaces between subdomains. An iterative method of an optimal order of arithmetical complexity is proposed for solving the arising algebraic systems in the case of regular quasiuniform hierarchical grids. An example of such a formulation was originally given in [Kuz95b]. The approach proposed here has many common points with the decentralization methods studied more than twenty years ago in [BLT74, Lem74]. In these papers the authors used splittings of bilinear forms between different subdomains to decompose a variational problem.

The second important topic is presented in Section 4 where we consider an extension of results from [Kuz95a, Kuz95b] to the case of overlapping subdomains. Here we present several results which mainly concern the construction of the interface preconditioner.

In Section 5 results of numerical experiments with 2D- and 3D-overlapping subdomains are given.

2 Macro-hybrid based on Overlapping Domain Decomposition

Let us consider a model elliptic problem

$$\begin{aligned} -\Delta u + cu &= f \quad \text{in } \Omega \\ \frac{\partial u}{\partial \mathbf{n}} &= 0 \quad \text{on } \partial\Omega \end{aligned} \tag{2.1}$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

where $f \in L_2(\Omega)$ is a given function, $c \equiv \text{const} \in (0;\, 1]$, $\partial\Omega$ is the boundary of a domain Ω and $\mathbf{n}$ is the outer unit normal vector to $\partial\Omega$. For the sake of simplicity we assume that Ω is a polygon in $\mathbf{R}^p$, $p = 2,\, 3$ with $\operatorname{diam}(\Omega) \sim O(1)$, and all further subdomains of Ω are also polygons with diameters $O(1)$.

The classical weak formulation of (2.1) is: find $u \in H^1(\Omega)$ such that

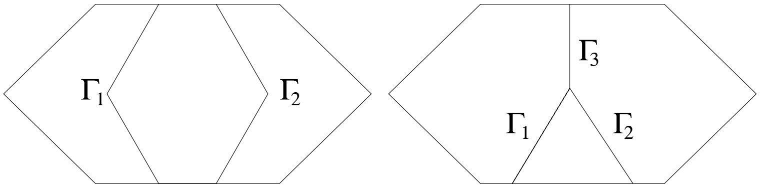
$$\Phi(u) = \min_{v \in H^1(\Omega)} \Phi(v), \tag{2.2}$$

where

$$\Phi(v) = \int\limits_{\Omega} \left[|\nabla v|^2 + cv^2 - 2fv \right] d\Omega. \tag{2.3}$$

Let Ω_1 and Ω_2 be two overlapping subdomains of Ω ($\Omega_1 \cap \Omega_2 \neq \emptyset$) such that $\overline{\Omega_1 \cup \Omega_2} = \overline{\Omega}$. We assume that subdomains Ω_1 and Ω_2 are regularly shaped. Examples of such a partitioning of Ω into two subdomains are given in Fig. 1 (Ω_1 is located on the left, i.e. $\Gamma_2 \subset \partial\Omega_1 \cap \Omega$).

Figure 1 a) Overlapping and b) overlapping / nonoverlapping domain decomposition



We denote the intersection of Ω_1 and Ω_2 by Ω_{12} and define two bilinear forms

$$a_k(u,\, v) = \int\limits_{\Omega_k} \left[a_k \nabla v \cdot \nabla u + c_k uv \right] d\Omega, \qquad k = 1,\, 2, \tag{2.4}$$

two linear forms

$$l_k(v) = \int\limits_{\Omega_k} f_k v d\Omega, \qquad k = 1,\, 2, \tag{2.5}$$

and two quadratic functionals

$$\psi_k(v) = a_k(v,\, v) - 2l_k(v), \qquad k = 1,\, 2. \tag{2.6}$$

The coefficients a_k, c_k and functions f_k are defined by

$$a_k = \begin{cases} 1 & \text{in } \Omega_k \setminus \Omega_{12} \\ q_k & \text{in } \Omega_{12} \end{cases} \qquad c_k = \begin{cases} c & \text{in } \Omega_k \setminus \Omega_{12} \\ q_k c & \text{in } \Omega_{12} \end{cases}$$
$$f_k = \begin{cases} f & \text{in } \Omega_k \setminus \Omega_{12} \\ q_k f & \text{in } \Omega_{12} \end{cases} \tag{2.7}$$

where q_k are positive constants, $k = 1, 2$ such that $q_1 + q_2 = 1$. It is important that

$$\psi_k(v) = q_k\Phi(v), \quad \forall v \in H^1(\Omega),\ \operatorname{supp} v \in \Omega_{12},\ k = 1, 2. \tag{2.8}$$

To introduce and to analyze macro-hybrid formulations of elliptic problems we have to deal with interfaces between subdomains. To this end we introduce the following notation:

$$\begin{aligned} \gamma_g &= (\partial\Omega_1 \cap \Omega) \cup (\partial\Omega_2 \cap \Omega), \\ \gamma_{in} &= \partial\Omega_1 \cap \partial\Omega_2, \\ \partial\gamma_{in} &= \bar{\gamma}_{in} \setminus \overset{\circ}{\gamma}_{in}\ . \end{aligned} \tag{2.9}$$

Here $\overset{\circ}{\gamma}_{in}$ is the interior part of γ_{in} with respect to $(p-1)$-dimensional topology, γ_g is said to be the "global" interface (with respect to the macro-hybrid formulation to be presented), and $\partial\gamma_{in}$ is the set of cross points in the case $p = 2$ and the set of interedges in the case $p = 3$.

The set $\gamma_g \setminus \partial\gamma_{in}$ can be presented as the union of nonoverlapping open subsets Γ_s, $s = 1, \dots, s_g$ such that each Γ_s is a piece-wise linear curve in the case of $p = 2$ and a piece-wise linear surface in the case of $p = 3$. It is obvious that this partitioning γ_g into nonoverlapping open subsets is unique.

For examples given in Fig. 1 we have $s_g = 2$ in the case a) and $s_g = 3$ in the case b). Respectively, $\partial\gamma_{in}$ consists of four points both in the cases a) and b).

Now we introduce the space $V = H^1(\Omega_1) \times H^1(\Omega_2)$, the space

$$W = \left\{\bar{v} = (v_1, v_2) :\ \bar{v} \in V,\ \int\limits_{\Gamma_s} (v_1 - v_2)\mu\, ds = 0,\ \forall\mu \in H^{-1/2}(\Gamma_s),\ s = 1, \dots, s_g\right\} \tag{2.10}$$

and the quadratic functional

$$\psi(\bar{v}) = \psi_1(v_1) + \psi_2(v_2), \quad \bar{v} \in V. \tag{2.11}$$

It can be shown (see, for instance [BF91]) that under the assumptions made the following macro-hybrid formulation of problem (2.1):

$$\bar{u} \in V :\ \psi(\bar{u}) = \min_{\bar{v}\in W} \psi(\bar{v}) \tag{2.12}$$

has a unique solution and is equivalent to problem (2.2). We understand the equivalence in the sense that

$$u(x) = u_k(x) \quad \forall x \in \Omega_k, \tag{2.13}$$

where u is the solution function to (2.2).

Problem (2.12) has also an equivalent formulation in terms of Lagrange multipliers. For instance, in the case of example in Fig. 1b it can be presented in the following form: find $(\bar{u}, \bar{\lambda}) \in V \times \Lambda$ such that

$$\begin{array}{lcl}
a_1(u_1, v_1) + \int\limits_{\Gamma_1} \lambda_1 v_1\, ds + \int\limits_{\Gamma_2} \lambda_2 v_1\, ds + \int\limits_{\Gamma_3} \lambda_3 v_1\, ds & = & l_1(v_1), \\
a_2(u_2, v_2) - \int\limits_{\Gamma_1} \lambda_1 v_2\, ds - \int\limits_{\Gamma_2} \lambda_2 v_2\, ds - \int\limits_{\Gamma_3} \lambda_3 v_2\, ds & = & l_2(v_2), \\
\int\limits_{\Gamma_1} (u_1 - u_2)\mu_1\, ds & = & 0, \\
\int\limits_{\Gamma_2} (u_1 - u_2)\mu_2\, ds & = & 0, \\
\int\limits_{\Gamma_3} (u_1 - u_2)\mu_3\, ds & = & 0,
\end{array} \tag{2.14}$$

$\forall(\bar{v}, \bar{\mu}) \in V \times \Lambda$. Here $\Lambda = \prod\limits_{s=1}^{3} H^{-1/2}(\Gamma_s)$. It can be easily shown that

$$\lambda_1 = q_2 \frac{\partial u_2}{\partial \mathbf{n_2}} \text{ on } \Gamma_1, \quad \lambda_2 = -q_1 \frac{\partial u_1}{\partial \mathbf{n_1}} \text{ on } \Gamma_2, \quad \lambda_3 = -\frac{\partial u_1}{\partial \mathbf{n_1}} \text{ on } \Gamma_3, \tag{2.15}$$

where $\mathbf{n_1}$ and $\mathbf{n_2}$ are the outer normal vectors to $\partial\Omega_1$ and $\partial\Omega_2$, respectively. Recall that $u_1 \equiv u$ in Ω_1 and $u_2 \equiv u$ in Ω_2.

In a compact form (2.14) can be presented [GW88, BF91] by: find $(\bar{u}, \bar{\lambda}) \in V \times \Lambda$ such that

$$\begin{array}{lcl}
\hat{a}(\bar{u}, \bar{v}) + b(\bar{\lambda}, \bar{v}) & = & \hat{l}(\bar{v}), \\
b(\bar{\mu}, \bar{u}) & = & 0, \qquad \forall(\bar{v}, \bar{\mu}) \in V \times \Lambda.
\end{array} \tag{2.16}$$

Here

$$\begin{array}{c}
V = \prod\limits_{k=1}^{m} V_k, \qquad V_k = H^1(\Omega_k),\ k = 1, \dots, m, \\
\Lambda = \prod\limits_{s=1}^{s_g} \Lambda_s, \qquad \Lambda_s = H^{-1/2}(\Gamma_s),\ s = 1, \dots, s_g, \\
\hat{a}(\bar{u}, \bar{v}) = \sum\limits_{k=1}^{m} a_k(u, v), \quad \hat{l}(\bar{v}) = \sum\limits_{k=1}^{m} l_k(v), \\
b(\bar{\mu}, \bar{u}) = \sum\limits_{s=1}^{3} \int (u_1 - u_2)\mu_s\, ds,
\end{array} \tag{2.17}$$

where $m = 2$ and $s_g = 3$.

Remark 1 *Generalization to a larger number of subdomains is straightforward. For instance, if we use the formulation (2.16)-(2.17) we have to assume that for any simply connected subdomain* $G \subset \Omega \setminus \gamma_g$ *a positive constant* q_G *exists such that*

$$a_k(u, v) = q_G a(u, v), \quad l_k(v) = q_G l(v), \quad \forall u, v \in H^1(\Omega), \quad \operatorname{supp} v \subset G, \quad k = \overline{1, m}. \tag{2.18}$$

Remark 2 *If $\int\limits_\Omega f\,d\Omega = 0$ and $c \ll 1$ then problem (2.1) can be considered as a singular perturbation of the Neumann problem*

$$\begin{array}{rcll} -\Delta u & = & f & \text{in } \Omega \\ \dfrac{\partial u}{\partial \mathbf{n}} & = & 0 & \text{on } \partial\Omega. \end{array} \tag{2.19}$$

3 The Mortar Element Method and Algebraic Systems

We consider the only case when Ω_{kh} are conforming triangular, $p = 2$, or tetrahedral, $p = 3$, partitions of Ω_k, $k = 1, \ldots, m$, and γ_g does not intersect the interiors of the grid cells. Then V_{kh} are the standard piecewise linear finite element subspaces of $V_k \equiv H^1(\Omega_k)$, $k = 1, \ldots, m$. The finite element subspaces $\Lambda_{sh} \subset \Lambda \equiv H^{-\frac{1}{2}}(\Gamma_s)$, $s = 1, \ldots, s_g$ are chosen using the mortar element technique from [BMP89, BM94, Kuz95b].

The mortar finite element discretization of (2.16)–(2.17) is defined by: find $(\bar{u}_h, \bar{\lambda}_h) \subset V_h \times \Lambda_h$ such that

$$\begin{array}{lcl} \hat{a}(\bar{u}_h, \bar{v}) + b(\bar{\lambda}_h, \bar{u}_h) & = & \hat{l}(\bar{v}), \\ b(\bar{\mu}, \bar{u}_h) & = & 0, \end{array}$$

$\forall\,(\bar{v}, \bar{\mu}) \in V_h \times \Lambda_h$ where $V_h = \prod\limits_{k=1}^{m} V_{kh}$ and $\Lambda_h = \prod\limits_{s=1}^{s_g} \Lambda_{sh}$. Problem (3.20) leads to an algebraic system

$$\mathcal{A}x = y \tag{3.20}$$

with a saddle-point matrix

$$\mathcal{A} = \begin{pmatrix} A & B^T \\ B & O \end{pmatrix} \tag{3.21}$$

and vectors

$$x = \begin{pmatrix} u \\ \lambda \end{pmatrix}, \qquad y = \begin{pmatrix} f \\ 0 \end{pmatrix}. \tag{3.22}$$

Here A is a symmetric positive definite matrix and $\operatorname{Ker} B^T = 0$. It follows immediately that $\det \mathcal{A} \neq 0$.

For further analysis we need a more detailed description of A and B in block forms. The simplest block representations of A and B^T are:

$$A = \begin{pmatrix} A_1 & O & O \\ O & \ddots & O \\ O & O & A_m \end{pmatrix}, \qquad B^T = \begin{pmatrix} B_1^T \\ \vdots \\ B_m^T \end{pmatrix} \tag{3.23}$$

Here the kth block corresponds to the degrees of freedom of the finite element space V_{kh}, $k = 1, \ldots, m$.

For each subdomain Ω_k we partition degrees of freedom (grid nodes) into two groups. In the second group denoted by γ we collect the degrees of freedom which correspond to the grid nodes belonging to $\gamma_k = \gamma_g \cap \bar{\Omega}_k$. All other degrees of freedom we collect in the first group denoted by I. These partitionings induce the following block representations:

$$A_k = \begin{pmatrix} A_{kI} & A_{kI\gamma} \\ A_{k\gamma I} & A_{k\gamma} \end{pmatrix}, \qquad B_k^T = \begin{pmatrix} O \\ B_{k\gamma}^T \end{pmatrix}. \tag{3.24}$$

Let $\mathcal{B}$ be a symmetric positive definite matrix and $\mathcal{H} = \mathcal{B}^{-1}$. Since $\mathcal{A} = \mathcal{A}^T$ the preconditioned Lanczos [MK74, Kuz95b] can be used to solve system (3.20). In this paper we also recommend the preconditioned conjugate method based on the $\mathcal{B}$-norm of minimal errors [MK74]:

$$\begin{aligned} \hat{p}_l &= \begin{cases} \mathcal{H}\xi^0, & l = 1, \\ \mathcal{H}\xi^{l-1} - \alpha_l \hat{p}_{l-1}, & l > 1, \end{cases} \\ p_l &= \mathcal{H}\mathcal{A}\hat{p}_l, \\ x^l &= x^{l-1} - \beta_l p_l, \end{aligned} \tag{3.25}$$

$$\alpha_l = \frac{(\xi^{l-1}, \mathcal{A}\hat{p}_{l-1})_{\mathcal{H}}}{(\mathcal{A}\hat{p}_{l-1}, \mathcal{A}\hat{p}_{l-1})_{\mathcal{H}}}, \qquad \beta_l = \frac{(\xi^{l-1}, \hat{p}_l)}{(\mathcal{A}\hat{p}_l, \mathcal{A}\hat{p}_l)_{\mathcal{H}}},$$

where $\xi^l = \mathcal{A}x^l - y$ are the residual vectors, $l = 1, 2, \ldots$ Assume that the eigenvalues of $\mathcal{H}\mathcal{A}$ belong to the union of segments $[d_1; d_2]$ and $[d_3; d_4]$ with $d_1 \le d_2 < 0 < d_3 \le d_4$. Then the convergence estimate

$$\|x^l - x\|_{\mathcal{H}} \le 2q^l \|x^0 - x\|_{\mathcal{H}}, \quad l \ge 1, \tag{3.26}$$

holds [MK74] where $q = \dfrac{\hat{d} - \check{d}}{\hat{d} + \check{d}}$, $\hat{d} = \max\{d_4; |d_1|\}$, and $\check{d} = \min\{d_3; |d_2|\}$.

4 Block Diagonal Preconditioning

We propose a preconditioner $\mathcal{H}$ as a block diagonal matrix:

$$\mathcal{H} = \begin{pmatrix} H_A & O \\ O & H_\lambda \end{pmatrix} \tag{4.27}$$

where H_A is also a block diagonal matrix:

$$H_A = \begin{pmatrix} H_1 & O & O \\ O & \ddots & O \\ O & O & H_m \end{pmatrix}. \tag{4.28}$$

All blocks are symmetric positive definite matrices. H_k are said to be the subdomain preconditioners, and H_λ is said to be the interface preconditioner.

If matrices H_k are spectrally equivalent to the matrices A_k^{-1} with constants independent of the value of the coefficient c, and if a matrix H_λ is spectrally equivalent to the matrix S_λ^{-1} with S_λ given by

$$S_\lambda = BA^{-1}B^T \equiv \sum_{k=1}^{m} B_{k\gamma} S_{k\gamma}^{-1} B_{k\gamma}^T \tag{4.29}$$

with the constants independent of the value of c then the values of $\hat{d}, \check{d}$ in (3.26) are positive constants [Kuz95a] also independent of c. Here

$$S_{k\gamma} = A_{k\gamma} - A_{k\gamma I} A_{kI}^{-1} A_{kI\gamma} \tag{4.30}$$

are the Schur complements. Our aim is to construct a preconditioner $\mathcal{H}$ spectrally equivalent [Kuz95a] to the matrix $\mathcal{A}^{-1}$ with constants independent of c.

Subdomain Preconditioners

Let us define matrices $\overset{\circ}{A}_k$ and M_k by:

$$\begin{aligned} (\overset{\circ}{A}_k\, v,\, w) &= \int_{\Omega_k} \nabla v_h \cdot \nabla w_h d\Omega, \\ (M_k v,\, w) &= \int_{\Omega_k} v_h w_h d\Omega \end{aligned} \tag{4.31}$$

$\forall\, v_h, w_h \in V_{kh}$, $k = 1, \ldots, m$. Thus, matrices $\overset{\circ}{A}_k$ are the stiffness matrices for the operator $-\Delta$ with the Neumann boundary conditions, and M_k are the corresponding mass matrices. It can be easily shown [Kuz95b] that

$$A_k^{-1} \sim \left(\overset{\circ}{A}_k + M_k\right)^{-1} + \frac{1}{c} P_k \tag{4.32}$$

where $P_k M_k$ is the M_k-orthogonal projector onto Ker $\overset{\circ}{A}_k$ and the sign "$\sim$" denotes the spectral equivalence. Moreover, the constants of the spectral equivalence in (4.32) are independent of the value of c.

Suppose that a matrix $\overset{\circ}{H}_k$ is spectrally equivalent to the matrix $\left(\overset{\circ}{A}_k + M_k\right)^{-1}$. Then the matrix

$$H_k = \overset{\circ}{H}_k + \frac{1}{c} P_k \tag{4.33}$$

is spectrally equivalent to matrix A_k^{-1} with constants independent of the value of c.

We have plenty of choices for $\overset{\circ}{H}_k$, $k = 1, \ldots, m$.

Interface Preconditioner

We can easily show [Kuz95b] that

$$S_{k\gamma}^{-1} \sim \tilde{S}_{k\gamma}^{-1} + \frac{1}{c} P_{k\gamma} \tag{4.34}$$

where $\tilde{S}_{k\gamma}^{-1}$ is the Schur complement for the matrix $\overset{\circ}{A}_k + M_k$ and $P_{k\gamma}M_{k\gamma}$ is the $M_{k\gamma}$ orthogonal projector onto $\operatorname{Ker} S_{k\gamma}$ in the case $c = 0$. Moreover, the constants of equivalence in (4.34) are independent of the value of c. Here $M_{k\gamma}$ is the interface mass matrix defined by:

$$(M_{k\gamma}v,\, w) \quad = \quad \int\limits_{\gamma_k} v_h w_h ds \quad \forall\, v_h, w_h \in V_{k\gamma h} \tag{4.35}$$

where $V_{k\gamma h}$ is the trace of V_{kh} into $\gamma_k = \partial\Omega_k \cap \Omega$, $k = 1, \dots, m$.

Let the matrices

$$\overset{\circ}{H}_k = \begin{pmatrix} \overset{\circ}{H}_{kI} & \overset{\circ}{H}_{kI\gamma} \\ \overset{\circ}{H}_{k\gamma I} & \overset{\circ}{H}_{k\gamma} \end{pmatrix} \tag{4.36}$$

be spectrally equivalent to the matrices $\left(\overset{\circ}{A}_k + M_k\right)^{-1}$, $k = 1, \dots, m$. We can also prove that the matrix

$$\hat{S}_\lambda = \sum_{k=1}^{m} B_{k\gamma}(\overset{\circ}{H}_{k\gamma} + \frac{1}{c}P_{k\gamma})B_{k\gamma}^T \tag{4.37}$$

is spectrally equivalent to S_λ with constants independent of the value of c.

To construct the interface preconditioner H_λ we shall use the preconditioned Chebyshev iterative procedure [BPS86, Kuz95a]. Let $\hat{H}_\lambda$ be a symmetric positive defined matrix and $\nu_\lambda = \lambda_{\max}/\lambda_{\min}$ where $\lambda_{\max}$ and $\lambda_{\min}$ are the maximal and minimal eigenvalues of $\hat{H}_\lambda\hat{S}_\lambda$, respectively. Then for any $t_\lambda \sim \sqrt{\nu_\lambda}$ the matrix

$$H_\lambda = \left[I_\lambda - \prod_{t=1}^{t_\lambda} \left(I_\lambda - \alpha_t \hat{H}_\lambda \hat{S}_\lambda \right) \right] \hat{S}_\lambda^{-1} \tag{4.38}$$

is spectrally equivalent to the matrix S_λ^{-1} where $\{\alpha_t\}$ is a set of the corresponding Chebyshev parameters.

Let $\hat{B}_\lambda$ be a symmetric positive definite matrix such that $1 \in [\mu_{\min}; \mu_{\max}]$ where $\mu_{\min}$ and $\mu_{\max}$ are the minimal and maximal eigenvalues of the matrix $\hat{B}_\lambda^{-1} \sum\limits_{k=1}^{m} B_{k\gamma} \overset{\circ}{H}_{k\gamma} B_{k\gamma}^T$, respectively. Then for the choice $\hat{H}_\lambda = \hat{R}_\lambda^{-1}$ where

$$\hat{R}_\lambda = \hat{B}_\lambda + \frac{1}{c}\sum_{k=1}^{m} B_{k\gamma}P_{k\gamma}B_{k\gamma}^T, \tag{4.39}$$

the estimate

$$\nu_\lambda \leq \hat{\nu}_\lambda \equiv \mu_{\max}/\mu_{\min} \tag{4.40}$$

holds.

A solution algorithm for a system

$$\hat{R}_\lambda z = g$$

is presented in [Kuz95b, KW95]. It includes a so called "coarse grid" problem based on the projectors $P_{k\gamma}$, $k = 1, \dots, m$.

Arithmetical Complexity for Hierarchical Grids

Assume that grids Ω_{kh} are regular, quasiuniform and hierarchical with the average grid step size $h \sim \sqrt[p]{N}$ where N is the dimension of matrix $\mathcal{A}$.

In this case we can use various V-cycle multilevel preconditioners to define matrix $\overset{\circ}{H}_k$ in (4.33). These preconditioners are spectrally equivalent to the matrices $(\overset{\circ}{A}_k +M_k)^{-1}$, $k = 1, \dots, m$ and have the optimal order of arithmetical complexity [Osw94, Xu92], i. e. the multiplication with such a preconditioner by a vector costs $O(N)$ arithmetical operations.

Our choice $\overset{\circ}{H}_{k\gamma}$ in (4.37) as the corresponding blocks of V-cycle multilevel preconditioner (BPX or MDS-type) is based on two observations. The first one is obvious: spectral equivalence of $\overset{\circ}{H}_{k\gamma}$ and $\tilde{S}^{-1}_{\gamma k}$ follows directly from the spectral equivalence of H_k and $\left(\overset{\circ}{A}_k +M_k\right)^{-1}$, $k = 1, \dots, m$. The second observation is rather technical and concerns implementation algorithms for V-cycle multilevel preconditioners: multiplication of $\overset{\circ}{H}_{k\gamma}$ by a vector can be implemented with $O(h^{1-p})$ arithmetical operations. The latter observation has at least one very important consequence: the corresponding matrix $\hat{S}_\lambda$ can be multiplied by a vector with $O(h^{1-p})$ arithmetical operations, i.e. multiplication with $\hat{S}_\lambda$ has the optimal order of arithmetical complexity.

It remains to choose preconditioner $\hat{R}_\lambda$, and we do not need an optimal preconditioner because the dimension of S_λ is much smaller than the dimension of A.

In paper [Kuz95a] we proposed to choose $\hat{B}_\lambda$ being equal to a scalar matrix which is a spectrally equivalent to the matrix $\sum\limits_{k=1}^{m} B_{k\gamma} M^{-1}_{k\gamma} B^T_{k\gamma}$. With this choice, obviously

$$\nu_\lambda \leq \text{const} \cdot h^{-2}$$

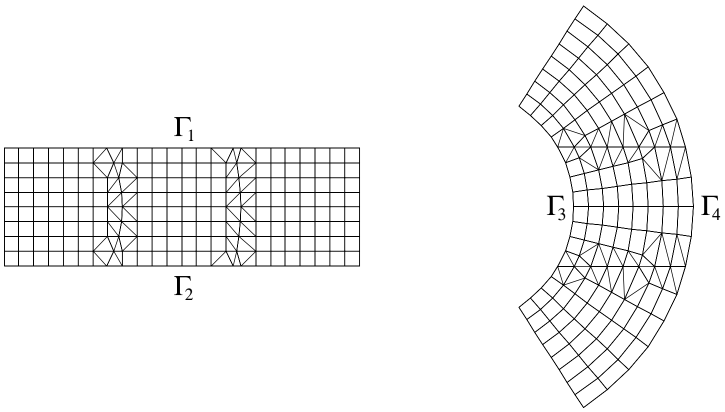
where the constant is independent of h and c, and the multiplication B^{-1}_λ by a vector can be implemented with $O(h^{1-p})$ arithmetical operations.

On the basis of the latter facts we conclude that t_ν should be proportional to h^{-1}, and arithmetical complexity of the corresponding preconditioner H_λ in (4.38) is of the order $O(h^{1-p})$. In some particular cases we can prove [BPS86, Kuz95a] that $t_\nu \sim h^{-1/2}$ and consequently the arithmetical complexity of H_λ is of the order $O(h^{1/2-p})$.

5 Numerical Experiment

The numerical experiments have been performed for two test cases.

Figure 2 Cartesian and polar locally fitted grids



The first test case is presented by the union of a rectangle and a segment of a ring. In the rectangular subdomain we have a rectangular cartesian grid and in the segment we have an orthogonal polar grid. Both grids are fitted to the interface boundary which consists of two straight segments Γ_1 and Γ_2, and two circle's segments Γ_3 and Γ_4. These grids are given in Fig. 2. Here $\partial\Omega_1 \cap \Omega = \Gamma_1 \cup \Gamma_2$ and $\partial\Omega_2 \cap \Omega = \Gamma_3 \cup \Gamma_4$.

Table 1 Cartesian / Polar grids

Cartesian grids in Ω_1	Polar grids in Ω_2	Number of Chebyshev iterations	Number of Lanczos iterations
24×8	16×8	13	68
48×16	32×16	22	72
96×32	64×32	32	75
192×64	128×64	45	77

The second test case is presented by the union of two parallelepipeds. The intersection of them is also a parallelepiped. Both grids are uniform and rectangular ones.

The results of numerical experiments are given in Tables 1 and 2. Two first columns contain information about the grids: product of numbers of nodes for each of the coordinates. The third column contains information about number of iterations t_λ used in construction of the interface preconditioner (4.38). The last column contains the number of iterations which were needed to reduce the $\mathcal{H}$-norm of the initial residual ξ^0 of 10^6 times by the preconditioned Lanczos method.

Table 2 Intersecting parallelepipeds: uniform grids

Grids in Ω_1	Grids in Ω_2	Number of Chebyshev iterations	Number of Lanczos iterations
$16 \times 16 \times 16$	$16 \times 8 \times 8$	24	54
$32 \times 32 \times 16$	$32 \times 16 \times 16$	38	51
$64 \times 64 \times 32$	$64 \times 16 \times 16$	52	51

One can see that t_λ grows up proportionally to $h^{-1/2}$ and that the number of Lanczos iterations is almost constant for both test cases.

Acknowledgement

The results of numerical experiments in Section 5 are kindly given for this paper by S. A. Finogenov and N. Vasiliev. The author is very grateful to both of them. The author would like also to thank P. Le Tallec and a reviewer for helpful discussions and comments.

REFERENCES

[BF91] Brezzi F. and Fortin M. (1991) *Mixed and Hybrid Finite Element Methods.* Springer–Verlag, New York.

[BLT74] Bensoussan A., Lions J.-L., and Temam R. (1974) Sur les methodes de decomposition, de decentralisation et de coordination et applications. In Lions J.-L. and Marchuk G. I. (eds) *Methodes Mathematiques de L'Informatique-4*, pages 133–257. Dunod, Paris.

[BM94] Ben Belgacem F. and Maday Y. (1994) The mortar element method for three dimensional finite elements. Technical report, Universite Paul Sabartier.

[BMP89] Bernardi C., Maday Y., and Patera A. (1989) A new nonconforming approach to domain decomposition: the mortar element method. In Lions J.-L. and Bresis H. (eds) *Nonlinear PDE and their Applications.* Pitman.

[BPS86] Bramble J., Pasciak J., and Schatz A. (1986) The construction of preconditioners for elliptic problems by substructuring, 1. *Math. Comp.* 47: 103–134.

[GW88] Glowinski R. and Wheeler M. (1988) Domain decomposition and mixed finite element methods for elliptic problems. In Glowinski R., Golub G., G.Meurant, and Periaux J. (eds) *Proc. First DD Int. Conf.*, pages 144–172. SIAM, Philadelphia.

[Kuz95a] Kuznetsov Y. A. (1995) Efficient iterative solvers for elliptic finite element problems on nonmatching grids. *Russ. J. Numer. Anal. Math. Modelling* 10: 187–211.

[Kuz95b] Kuznetsov Y. A. (June 1995) Iterative solvers for elliptic finite element problems on nonmatching grids. In *Proc. Int. Conf. AMCA-95*, pages 64–76. NCC Publisher, Novosibirsk.

[KW95] Kuznetsov Y. and Wheeler M. (1995) Optimal order substructuring preconditioners for mixed finite element methods on nonmatching grids. *East-West J. Numer. Math.* 3: 127–143.

[Lem74] Lemonnier P. (1974) Resolution numerique d'equations aux derivees partielles par decomposition et coordination. In Lions J.-L. and Marchuk G. I. (eds) *Methodes Mathematiques de L'Informatique-4*, pages 259–299. Dunod, Paris.

[MK74] Marchuk G. I. and Kuznetsov Y. A. (1974) Methodes iteratives et fonctionnelles quadratiques. In Lions J.-L. and Marchuk G. I. (eds) *Methodes Mathematiques de L'Informatique-4*. Dunod, Paris.

[Osw94] Oswald P. (1994) *Multilevel Finite Element Approximations: Theory and Applications*. Teubner-Scripten zur Numerik.

[Xu92] Xu J. (1992) Iterative methods by space decomposition and subspace correction. *SIAM Rev.* 34: 581–613.

73

The EAFE Scheme and CWDD Method for Convection-dominated Problems

Jinchao Xu

1 Introduction

In this paper, we derive a monotone finite element scheme for convection diffusion equations and then discuss a special domain decomposition method for the solution of the resulting algebraic system. The work was partially motivated by Brezzi, Marini and Pietra [BMP89] and Markowich and Zlamal [MZ89] where a Scharfetter-Gummel type ([SG69]) of finite element scheme was derived for symmetric positive definite equations in two spatial dimensions. The finite element scheme in this paper is derived by a completely novel technique for a very general class of convection diffusion equations (see (2.1) below). Some error estimates like those in [MZ89] can be obtained in a very straightforward fashion under the new derivation (see Xu and Zikatanov [XZed]). This finite element scheme is monotone for some very general (unstructured) meshes such as Delauney triangulations in two dimensions. A monotone finite element scheme is important not only from the viewpoint of better stability and approximation but also from the fact that the resulting linear algebraic system may be more effectively solved. In this paper, we also discuss an efficient domain decomposition method for the resulting scheme.

We would like to point out that there is a vast literature on numerical methods for convection dominated problems and many special techniques have been developed, to name a few, for example, we refer to [BBFS90], [BMP89], [DEO92], [Hug95], [Joh87], [RST96] and the references cited therein. It is fair to say that all the different methods in the aforementioned papers are related in certain sense. In particular, the monotone schme described in this paper is also essentially similar to many other monotone schemes in the literature. But we emphasis that our scheme is derived in a quite different and elegant manner and it can be neatly applied to rather general unstructured grids in any spatial dimensions.

The rest of the paper is organized as follows. In we discuss a model problem and some properties of finite element discretization for the Poisson equation. In we derive

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

an edge-average finite element scheme and discuss the monotonicity property of the scheme. In , we present some numerical example and discuss a *cross-wind strip* domain decomposition technique. We make some concluding remarks in .

2 Model Problems and Finite Element Spaces

We shall mainly consider the following model convection diffusion problem:

$$Lu \equiv -\nabla \cdot (\alpha(x)\nabla u + \beta(x)u) + \gamma(x)u = f(x) \; x \in \Omega \quad \text{and} \quad u = 0 \; x \in \partial\Omega \tag{2.1}$$

where Ω is a polygonal domain in $R^n (n \geq 1)$ with boundary $\partial\Omega$, $f \in L^2(\Omega)$. We assume that α, β and γ are bounded and piecewise smooth functions and $\alpha(x) \geq \alpha_{min} > 0, \gamma(x) \geq 0$ for every $x \in \Omega$.

The weak formulation of the problem (2.1) is: Find $u \in H_0^1(\Omega)$ such that

$$a(u, v) = f(v) \qquad \text{for all } v \in H_0^1(\Omega), \tag{2.2}$$

where

$$a(u, v) = \int_\Omega (\alpha(x)\nabla u + \beta(x)u) \cdot \nabla v + \gamma(x)uv \, dx, \qquad f(v) = \int_\Omega f(x)vdx. \tag{2.3}$$

It is well-known that (2.2) is uniquely solvable for any $f \in L^2(\Omega)$. Furthermore L^{-1} is a positive operator, namely (see Gilbarg and Trudinger [GT83])

$$(L^{-1}f)(x) \geq 0 \text{ for all } x \in \Omega, \text{ if } f(x) \geq 0 \text{ for all } x \in \Omega. \tag{2.4}$$

The above condition will be loosely referred as the *monotonicity property.*

We are interested in the convection dominated case, namely $\beta(x)/\alpha(x) \gg 1$ for all $x \in \Omega$. For the convection dominated case, it is well known that a standard finite element discretization scheme will not give satisfactory result (in fact the discrete solution will have a lot of oscillations), but a scheme possessing a discrete monotonicity property like (2.4) would have much better approximation and stability properties. One aim of this paper is to derive such kind of finite element scheme. To be more specific, if $V_h \subset H_0^1(\Omega)$ is a finite element space and L_h is the corresponding discretized operator for L given in (2.1), we are looking for special finite element scheme such that

$$(L_h^{-1}f_h)(x) \geq 0 \text{ for all } x \in \Omega, \text{ if } f_h \in V_h \text{ and } f_h(x) \geq 0 \text{ for all } x \in \Omega. \tag{2.5}$$

A finite element scheme satisfying the above condition will be known as *a monotone* finite element scheme in this paper.

M-matrix property of the stiffness matrix for Poisson equation As we shall see late that our monotone finite element scheme is closely related to the Poisson equation:

$$-\Delta u = f(x) \; x \in \Omega \quad \text{and} \quad u = 0 \; x \in \partial\Omega. \tag{2.6}$$

For completeness, we now give some details concerning the finite element discretization for this simple equation. In particular we discuss a necessary and sufficient geometric condition for the stiffness matrix to be an M-matrix.

Let $\mathcal{T}_h$ be a usual finite element triangulation of Ω consisting of simplices and $V_h \subset H_0^1(\Omega)$ be the corresponding finite element space consisting of continuous piecewise linear functions. Given an element $\tau \in \mathcal{T}_h$ with vertices q_j $(1 \leq j \leq n+1)$, let (a_{ij}^{τ}) be the element stiffness matrix of the Poisson equation, namely $(\nabla u_h, \nabla v_h)_\tau = \sum_{i,j} a_{ij}^{\tau} u_h(q_i) v_h(q_j)$. Noting that $a_{ii}^{\tau} = -\sum_{j \neq i} a_{ij}^{\tau}$, we deduce that

$$(\nabla u_h, \nabla v_h) = \sum_{\tau \in \mathcal{T}_h} \sum_{\mathrm{E} \subset \tau} \omega_{\mathrm{E}}^{\tau} \delta_{\mathrm{E}} u_h \delta_{\mathrm{E}} v_h. \tag{2.7}$$

where $\omega_{\mathrm{E}}^{\tau} = -a_{ij}^{\tau}$ with E connecting the vertices q_i and q_j and $\delta_{\mathrm{E}} = u_h(q_i) - u_h(q_j)$.

For $i \neq j$, let $|\kappa_{ij}|$ denote the measure of $n-2$ dimensional simplex, κ_{ij}, opposite to the edge $\mathrm{E} = (q_i, q_j)$ and θ_{ij} is the angle between the two $n-1$ dimensional simplexes whose intersection forms κ_{E}. Then it can be proved that (see Barth [Bar92] for the case $n = 3$)

$$a_{ij}^{\tau} = -\frac{1}{n(n-1)} |\kappa_{\mathrm{E}}| \cot \theta_{\mathrm{E}}. \tag{2.8}$$

Consequently the stiffness matrix for the Poisson equation is an M-matrix if and only if the for any fixed edge E the following condition holds:

$$\sum_{\tau \supset \mathrm{E}} |\kappa_{\mathrm{E}}| \cot \theta_{\mathrm{E}} \geq 0, \tag{2.9}$$

where $\sum_{\tau \supset \mathrm{E}}$ means the sum over all simplices containing E.

We would like to remark that, in two dimensions, the condition (2.9) means that $\mathcal{T}_h$ is a *Delauney triangulation* in the sense that the sum of any the two angles opposite to any edge is less than or equal to π.

3 Derivation of a Monotone Finite Element Scheme

In this section, we give a derivation of a monotone finite element scheme for convection diffusion equation (2.1). Unlike many other schemes, this scheme does not make explicit reference to the flow (convection) directions.

Given a triangulation $\mathcal{T}_h$ as described in previous section, let us introduce a function $\psi_{\mathrm{E}}(s)$ defined locally over any fixed edge $\mathrm{E} = (q_i, q_j)$ in $\mathcal{T}_h$ by

$$\frac{\partial \psi_{\mathrm{E}}}{\partial \tau_{\mathrm{E}}} = \frac{1}{|\tau_{\mathrm{E}}|} \alpha^{-1} (\beta \cdot \tau_{\mathrm{E}}). \tag{3.10}$$

Here $\tau_{\mathrm{E}} = q_i - q_j$ and $\frac{\partial}{\partial \tau_{\mathrm{E}}}$ denotes the tangential derivative along E.

Set $J(u) = \alpha \nabla u + \beta u$. Multiplying $J(u)$ by α^{-1}, taking the Euclidean inner product with the directional vector τ_{E} and using the definition of ψ_{E} in (3.10) we obtain:

$$e^{-\psi_{\mathrm{E}}} \frac{\partial (e^{\psi_{\mathrm{E}}} u)}{\partial \tau_{\mathrm{E}}} = \frac{1}{|\tau_{\mathrm{E}}|} \alpha^{-1} (J(u) \cdot \tau_{\mathrm{E}}). \tag{3.11}$$

After integration over edge E, we obtain the following identity:

$$\delta_{\mathrm{E}}(e^{\psi_{\mathrm{E}}}u) = \frac{1}{|\tau_{\mathrm{E}}|}\int_{\mathrm{E}} \alpha^{-1}e^{\psi_{\mathrm{E}}}(J(u)\cdot\tau_{\mathrm{E}})ds. \tag{3.12}$$

The crucial step of our derivation is to assume that $J(u)$ is approximated by a constant vector, say $J_\tau(u)$, over each simplex τ. Once this (only) extra assumption is made, then from (3.12), this constant vector may be related by

$$J_\tau(u)\cdot\tau_{\mathrm{E}} \approx \tilde{a}_{\beta,\mathrm{E}}\delta_{\mathrm{E}}(e^{\psi_{\mathrm{E}}}u), \qquad \tilde{a}_{\beta,\mathrm{E}} = \left[\frac{1}{|\tau_{\mathrm{E}}|}\int_{\mathrm{E}}\alpha^{-1}e^{\psi_{\mathrm{E}}}ds\right]^{-1}. \tag{3.13}$$

Now using the representation from (2.7) we have the following identity for $\phi_h \in V_h$

$$\int_\tau J_\tau(u)\cdot\nabla\phi_h = \sum_{\mathrm{E}}\omega_{\mathrm{E}}^\tau(J_\tau(u)\cdot\tau_{\mathrm{E}})\delta_{\mathrm{E}}\phi_h. \tag{3.14}$$

This leads to the following modified bilinear form:

$$a_h(u_h,\phi_h) = \sum_{\tau\in\mathcal{J}_h}\left\{\sum_{\mathrm{E}\subset\tau}\omega_{\mathrm{E}}^\tau\tilde{a}_{\beta,\mathrm{E}}\delta_{\mathrm{E}}(e^{\psi_{\mathrm{E}}}u_h)\delta_{\mathrm{E}}\phi_h + \gamma_\tau(u_h\phi_h)\right\}, \tag{3.15}$$

where a lumped mass quadrature rule is used for the zero order term in $a(\cdot,\cdot)$:

$$\gamma_\tau(u\phi_h) = \frac{|\tau|}{n+1}\sum_{i=1}^{n+1}\gamma(q_i)u(q_i)\phi_h(q_i).$$

The resulting finite element discretization is: Find $u_h \in V_h$ such that

$$a_h(u_h,\phi_h) = f(\phi_h) \qquad \text{for all } \ \phi_h \in V_h. \tag{3.16}$$

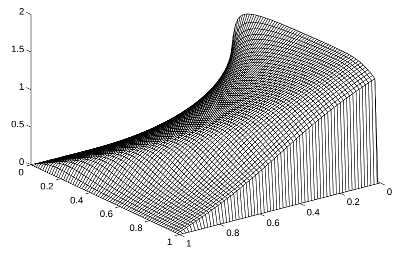
The above derivation appears quite simple, but the resulting scheme has a remarkable (but obvious) monotonicity property.

Lemma 3.1 *The stiffness matrix corresponding to the bilinear form (3.15) is an M-matrix for any $\alpha > 0$, $\gamma \geq 0$ and β, if and only if the stiffness matrix for the Poisson equation is an M-matrix, namely if and only if the condition (2.9) holds.*

This means that our finite element scheme (3.16), regardless the behavior of the convection coefficient β, satisfies a desirable monotonicity property for a very general class of meshes such as Delauney triangulations in two dimensions. Note that our derivation makes little use of any specific property of β. In the very special case that $\beta/\alpha = \nabla\psi$ for some function ψ (which means that (2.1) is symmetrizable) and $n = 2$, the scheme (3.16) is reduced to the Scharfetter-Gummel scheme derived in [MZ89], although our derivation technique is completely different.

We would like remark that the exponential functions in the (3.16) would not cause any numerical problems if they are handled with caution. For detailed discussions, we refer to [XZed] and [WX].

Figure 1 Surface plot of the discrete solution to (4.18).



Formal convergence rate It is well known that monotone schemes can only have first order accuracy in general. It can be easily shown that our scheme admits the following *formal* error estimate:

$$|u_I - u_h|_{H^1(\Omega)} \leq Ch(|J(u)|_{W^{1,p}(\Omega)} + |\gamma u|_{W^{1,p}(\Omega)}, \tag{3.17}$$

where u_I is the usual finite element interpolant of the solution of the problem (2.1) and $p > n$. (Details of this error analysis are reported in [XZed]). The error estimate (3.17) is formal since the constant C and the norm $|J(u)|_{1,p,\Omega}$ there depend on α and β. It is interesting to note that this monotone finite element scheme gives a first order (formal) accuracy in H^1 norm.

4 Numerical Examples and the CWS Method

In this section we first give a simple but not trivial example of convection dominated problem to demonstrate the behavior of our finite finite element scheme and then briefly discuss a domain decomposition strategy.

First numerical example We consider the following test problem:

$$-\nabla \cdot (\varepsilon \nabla u + (y, -x)u) = 1, \tag{4.18}$$

subject to the homogeneous Dirichlet boundary conditions on unit square $(0,1) \times (0,1)$. Note that the partial differential operator here is not symmetrizable.

Shown in Figure 1 is a finite element solution obtained on a uniform triangular grid on the unit square with $\varepsilon = 10^{-6}$ and $h = 2^{-6}$ ($h/\varepsilon = 15625$). The quality of the numerical solution looks quite good and no spurious oscillations or smearing near the boundary layer are observed.

A cross-wind strip domain decomposition strategy It is known that monotone schemes may be solved effectively by Gauß-Seidel methods if the unknowns are properly ordered see, for example, [BY92], [EC93], [Far89], [BWnt]. But for unstructured grids, an optimal ordering can be very difficult to realize. Here we give a brief discussion of a special domain decomposition strategy that proves to

Figure 2 Crosswind blocking with $\beta = (y, -x)$ and an unstructured grid.

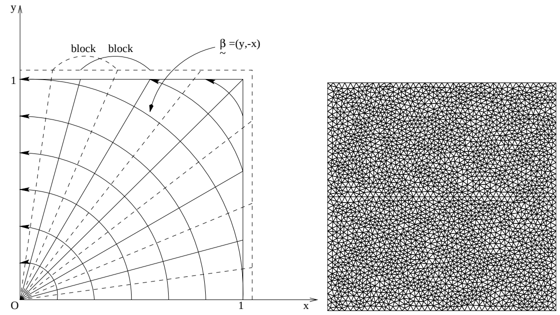


Table 1 Performance of CWS method for (4.18 on unstructured meshes

ε	h	#nodes	CWS method	
			iteration count	CPU time (seconds)
10^{-5}	1/20	729	1	0.004
	1/40	2839	2	0.27
	1/80	11193	3	1.7

be quite effective for unstructured grids with variable convections. More details of this algorithm can be found in Wang and Xu ([WX]).

We here describe the strategy for the two dimensional case. The idea is to decompose the computational domain into very thin strips which are orthogonal to the convection direction given by β. Figure 2 illustrates the case for $\beta = (y, -x)$ as in the previous numerical example. Corresponding to each of these strips, a subdomain can be defined by the union of supports of the basis functions associated with the nodes belonging to the strip. We then carry out the multiplicative Schwarz or successive subspace correction ([Xu92]) domain decomposition method to the aforementioned domain decomposition (possibly with overlapping) with an order following the direction of β. We use banded Gaussian elimination on each subdomain since the strip is very thin and the stiffness matrix associated with each subdomain is a banded matrix with a very small bandwidth. We call this kind of method to be the *cross-wind strip* (CWS) domain decomposition method.

Table 1 shows the performance of CWS method for problem (4.18) discretized by our monotone finite element scheme on unstructured grids on the unit square. The computation was carried on an DEC Alpha-Workstation 5000/240. The iteration stops when the maximum norm of residual vector reaches 10^{-8}. The mesh size h shown in the table is the characteristic size of each mesh. The right hand of plot in Figure 2 is the unstructured grid with characteristic size $h = 1/40$. As we see that the CWS converges extremely fast.

5 Concluding Remarks

The EAFE (edge-average finite element) scheme derived in this paper proves to be an effective approach to discretizing convection dominated problems. The derivation of this scheme is simple and is valid in all spatial dimensions. The monotonicity property of the scheme is uniformly valid for any feasible diffusion and convection coefficients and any size of meshes (as long as satisfying some mild geometric constraints such as being Delauney triangulations in two dimensions).

The CWS (cross-wind strip) domain decomposition method provides a very efficient approach to solving the algebraic systems resulting from a monotone scheme. Instead of using elaborative techniques for ordering the unknowns, the CWS method makes more but simple use of geometric property of the underlying meshes in association with convection coefficients. For strongly convection dominated problems, this method converges in very few iterations. For mildly convection dominated problems or for problems with sharply variable convections, the CWS method may be used as an effective smoother in a multigrid process (see [WX]).

Like any other schemes for convection dominated problems, the convergence analysis for EAFE method is a very technical task. The theoretical justification of the efficiency of CWS method for general unstructured grids is also lacking. These theoretical issues will be addressed in our future work. More importantly, our main goal is to develop efficient multigrid methods and some satisfactory corresponding theory for convection dominated equations and hyperbolic problems in general, which in fact is the main motivation of the current work.

Acknowledgement

The author wishes to thank David Keyes and Tony Chan for their hospitality and discussions during the author's visits at ICASE (September 1995) and UCLA (Fall 1995 - Spring 1996) while this work was carried out. Thanks also go to Randy Bank and Ludmil Zikatanov for helpful discussions on the relevant topics and Feng Wang for his help on numerical experiments. The author is partially supported by NSF DMS94-03915-1 through Penn State and by NSF ASC-92-01266 and ONR-N00014-92-J-1890 through UCLA.

REFERENCES

[Bar92] Barth T. (1992) Aspects of unstructured grids and finite-volume solvers for the Euler and Navier-Stokes equations. Technical Report AGARD Report 787, AGARD. Speical course on unstructured grids methods for advection dominated flows.

[BBFS90] Bank R., Bürger J., Fichtner W., and Smith R. (1990) Some up-winding techniques for finite element approximations of convection diffusion equations. *Numer. Math.* 58: 185–202.

[BMP89] Brezzi F., Marini L., and Pietra P. (1989) 2-dimensional exponential fitting and applications to drift-diffusion models. *SIAM J. Num. Anal.* 26(6): 1342–1355.

[BWnt] Bey J. and Wittum G. (preprint) Downwind numbering: A robust multigrid method for convection-diffusion problems on unstrustured grids.

[BY92] Brandt A. and Yavneh I. (1992) On multigrid solution of high-Reynolds incompressible entering flows. *J. Comput. Phys.* 101: 151–164.

[DEO92] Dorlofsky L., Engquist B., and Osher S. (1992) Triangle based adaptive stencils for the solution of hyperbolic conservation laws. *J. Comp. Phys.* 98(1).

[EC93] Elman H. C. and Chernesky M. P. (1993) Ordering effects on relaxation methods applied to the discrete one-dimensional convection-diffusion equation. *SIAM J. Numer. Anal.* 30(5): 1268–1290.

[Far89] Farrell P. A. (1989) Flow conforming iterative methods for convection dominated flows. In *Numerical and Applied Mathematics*, volume 1.2 of *IMACS annals on computing and applied mathematics*, pages 681–686. J. C. Baltzer AG, Scientific Co., Basel, Switzerland.

[GT83] Gilbarg D. and Trudinger N. (1983) *Elliptic Partial Differential Equations of Second Order.* Springer-Verlag.

[Hug95] Hughes T. (1995) Multiscale phenomena: Greens functions, the dirichlet-to-neumann formulation, subgrid scale models, bubles and the origins of stabilized methods. *Comp. Meth. in Appl. Mech. Eng.* 127: 387–401.

[Joh87] Johnson C. (1987) *Numerical Solution of Partial Differential Equations by the Finite Element Method.* Cambridge University Press, Cambridge.

[MZ89] Markowich P. and Zlamal M. (1989) Inverse-average-type finite element discretizations of self-adjoint second order elliptic problems. *Math. Comp.* 51: 431–449.

[RST96] Roos H., Stynes M., and Tobiska L. (1996) *Numerical methods for singularly perturbed differential equations.* Springer.

[SG69] Scharffetter D. and Gummel H. (1969) Large-signal analysis of a silicon read diode oscillator. *IEEE Trans. Electron Devices* ED-16: 64–77.

[WX] Wang F. and Xu J.A crosswind domain decomposition method for convection dominated problems. To be submitted to Siam J. on Scientific Computing.

[Xu92] Xu J. (1992) Iterative methods by space decomposition and subspace correction. *SIAM Review* 34: 581–613.

[XZed] Xu J. and Zikatanov L. (submitted) A monotone finite element scheme for convection diffusion equations. *Math. Comp.* .

Part III

Implementation tools

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

74

A Computational Environment based on a Domain Decomposition Approach

Edgar A. Gerteisen and Ralf Gruber

1 Introduction

With increasing performance of computational facilities, the complexity of geometries being tackled by scientists and engineers has grown constantly. Simultaneously, the generation of appropriate meshes has become a subject of increasing interest. One popular approach to discretize the physical space is the use of structured mesh blocks which themselves are combined in an unstructured fashion. In FE terminology this approach can be classified as nonoverlapping structured mortar elements with matching grid points. Although domain decomposition (DD) methods recently have received significant attention because of their natural route to be mapped onto distributed memory architectures, the main benefit of block-structured techniques arises from combining the advantages of both structured and unstructured approaches. Interactive tools have been proposed which assist in defining appropriate block topologies [Sei86, Con], though the difficulty and the amount of engineering time for topology generation can become considerable. Simultaneously, the interactive approach militates somewhat against automatization and embedding into parameter optimization procedures.

The present contribution first describes the parametric representation and the corresponding automatized topology generation for a composed complex shape. The system used is a graphically assisted and command language based, which can be regarded as a mixture between fully automatic and purely interactive. Consequently, it can be used to prototype the automatization of specific geometry classes or for

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

combining predefined classes that themselves are already completely automatized or unstructured representations of the computational space. Aside from the mesh generation basic generic data structures (DS) are required enabling for communication of iterative numeric schemes. The application behind the present study is the computational model of an electrostatic precipitation (ESP) process, which is used in a variety of technical processes to eliminate particles from exhaust gas. Some of the important physical properties are the electric field, the space charge, the flow velocity field, the particle charges, and particle sizes and the velocities of different flow phases [EKG97], all of which are nonlinearly coupled.

2 Computational Environment

The computational environment is a Problem-Solving Environment that aims at the realization of a system for cooperative design and development. It can be considered as a sort of skeleton for embedding applications supplemented by additional, not yet available or alternative modules introduced by well defined interfaces [GEG$^+$95]. Such an open system is especially demanded for multi-disciplinary applications, in which knowledge of specialists in different fields has to be combined [EKG97]. A data driven approach is required for realizing a transparent interaction between different modules and consequently the usage of a common database is an essential part (see Figure 1) [Mer95]. Data flow between modules is realized via the database and controlled by an application-dependent software layer that can be implemented by means of a command language driven user interface. The information system consists of a database monitor and a 3D graphics package, and eventually of additional application-specific postprocessing modules.

3 Geometry

The geometry under consideration consists of a channel (here two planar plates) with tube-like objects placed at different distances from the channel walls (here center plane) along the main channel direction (Figure 3). The tube-like objects have prongs along the tube axe with different angular orientation, the prongs themselves have teeth which again are defined by several parameters (see Figure 2). Parameters include the channel width, the position of the objects within the channel, and the parameters for the tube itself, e.g., diameter of tube, length of prongs, height of prongs, angle between prongs, etc.

The geometry definition procedure includes several aspects. First, the geometric surfaces need to be constructed or a given CAD description has to be transferred into a representation suited for generating a computational mesh. In general, CAD surfaces have to be reconstructed (condensation, closing of gaps, etc.), which can, in principle, be carried out with arbitrary high geometry fidelity by means of a point-wise projection. Additional geometric functionalities are required for generating a parametric topology and for defining boundary conditions fulfilling a complete and proper problem description.

Figure 1 Program environment for the ESP application. Starting from a skeleton that includes basic functionalities and together with already existing modules, additional functionalities are added to form the basis for an ESP simulation environment.

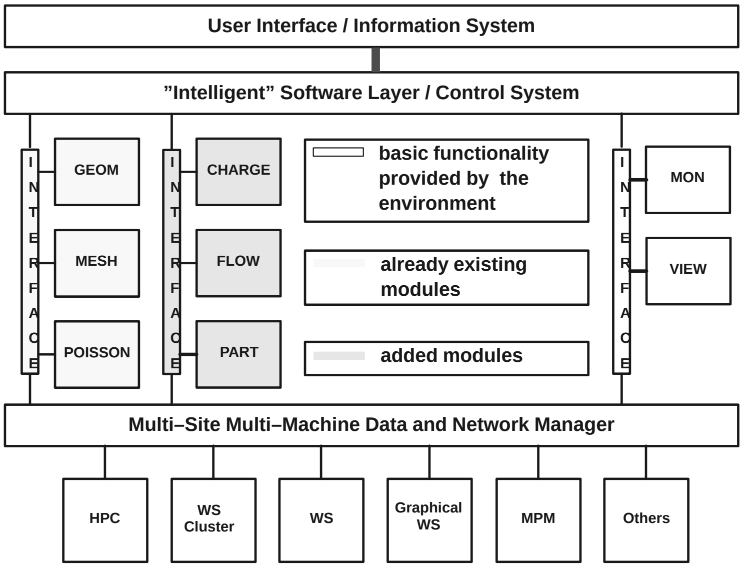
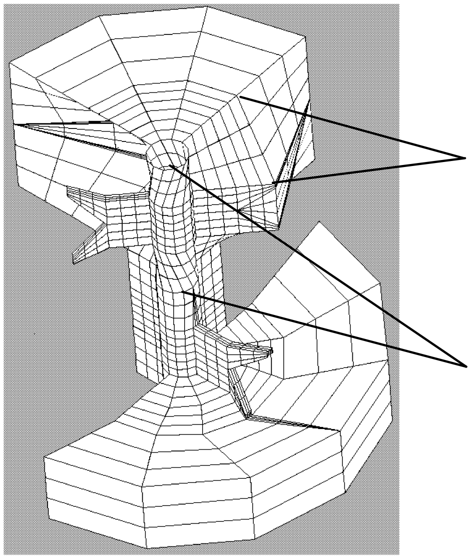

User Interface / Information System
"Intelligent" Software Layer / Control System
INTERFACE
GEOM
MESH
POISSON
INTERFACE
CHARGE
FLOW
PART
basic functionality provided by the environment
already existing modules
added modules
INTERFACE
MON
VIEW
Multi–Site Multi–Machine Data and Network Manager
HPC
WS Cluster
WS
Graphical WS
MPM
Others

Figure 2 Part of block-structured mesh topology consisting of 380 mesh blocks (each with 3×3 cells in this illustration) demonstrating the representation of arbitrary angles between prongs in tube direction, here 135°.



building block in tube direction which increments γ by 35°

building block in tube direction which increments γ by 90°

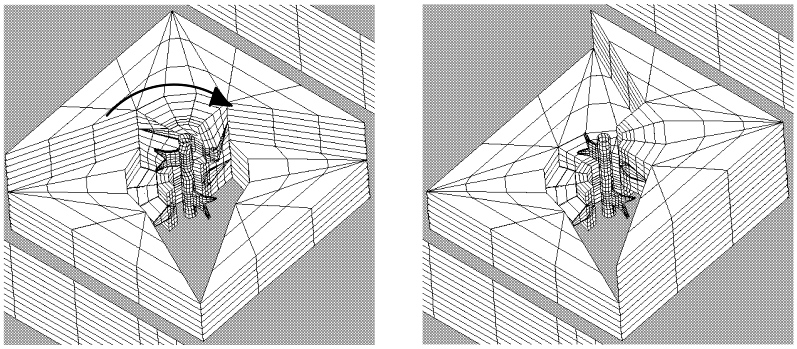
Geometry Surface Description

The basic elements of the command language are described in [MBF+90]. CAD primitives used for the surface description are create point (cp) and create face (CF). The create point function provides the basic functionality of defining CAD points and the create face function enables to generate basic CAD surfaces, which are defined by a structured set of CAD points in the two index directions. Additional important functionalities of the script language are control structures, such as “while” and branching by “if” conditions. It should be mentioned here that parts of the topology introduced afterwards, specifically the building blocks, need to be anticipated already during the construction of surface patches.

Topology Generation

First of all a strategy for the topology design has to be developed. The basic geometry is a rectangular channel. Several of the complicated tube-like structures (Figure 2) need to be fitted into this base configuration for which hexahedron building blocks are used (Figure 3). Inside those elementary geometric objects a transition to a circular tube is defined such that a quantum movement in circular direction of $\gamma = 90°$ is rendered possible, simultaneously maintaining matching grid points. The elementary object is divided further into four substructures in tube direction, two of them for allowing an arbitrary angle between prongs and one for each prong tooth (Figure 2).

Figure 3 Part of the mesh showing the hexahedron building block around tube-like structure. Identical domain numbers are activated and the angle variation between the prongs are causing a movement of the upper part by 90°. Angle of prongs: left picture = 135°, right picture = 180°.



Volumetric objects are created by defining opposite surfaces, again by cp and CF commands, for each domain. The CSD, create structured domain, command defines a domain connected by two faces. The structured domain is a CAD element and should not be mistaken with the structured mesh. A more comprehensive description of the outlined steps is given in [Ger96]. For the given example, with two tube elements and two prongs in tube direction activated, a topology consisting of 380 domains is generated. This number can easily reach the order of several thousand upon parameter variation.

Boundary Conditions

An important issue is the introduction of boundary conditions (BCs). In fact, the topology may vary upon parameter variation (see Figure 3) and the definition of BCs need to vary correspondingly. Therefore, geometric attributes (referring to inflow, outflow, solid wall, etc.) are placed on each CAD surface, which are transferred to the structured mortars and interpreted afterwards by the computational modules. The BCs are introduced within CSD command. The boundary-condition-on-face commands are embedded in control structures, since they are supposed to branch upon parameter variation. Additional modules exist for checking the completeness of the problem definition, e.g., no disjoint internal surface patches exist. All free surface patches are furnished with proper boundary condition codes.

4 Mesh

The computational mesh is derived by applying curvilinear interpolation within each CAD volume separately. The mesh module allows also for adaptive mesh generation based on a mesh density function [Bon90]. This monitor function is, in general, defined separately in the discrete space and it resides as an object in the database. It can be based on a combination of physical properties or on estimates for the discretization error of a numerical scheme. The adaptation algorithm is based on equidistribution principle and it applies a one-dimensional r-refinement (redistribution of mesh points maintaining the structure) along each index direction of a structured mesh block similar to the work of [DKS80]. However, regularization concepts [And87, CAA87] need to be introduced, in order to avoid skewed meshes and discontinuities of the mesh lines at block interfaces in the first derivatives.

5 Solver Modules

According to the multi-disciplinarity of the considered problem, diverse computational modules are included in the environment, e.g., finite element, finite volume, particle transport, characteristic method, each of them requiring a different type of communication. The cell-centered finite volume flow solver is based on a data structure of overlapping cells, whereas a master/slave DS appears to be better suited for nodal based schemes [Ger94]. The communication matrices are derived by a separate module and saved in the database for the sake of modularity. Alternatively they can be produced by calls to a specific library.

One example is the finite element solver module that, in the present application, is used for the computation of the electrical potential. The iterative conjugate gradient solver has been reimplemented recently to allow for an efficient usage of latest computer architectures. The system matrix is not completely assembled at the mortar element interfaces. However, each interface point is treated either as a master or as a slave and together with the corresponding communication structure the global matrix vector multiplication and scalar product can be computed properly. Those are implemented in form of a communication library that can be ported readily to different hardware architectures. Results on the NEC-SX4 indicate good performance on emerging vector based parallel architectures, namely 0.9 Gflop/s on one and up to 2.7 Gflop/s on four processing elements of the SX4 vector multiprocessor system, with a minor parallelization effort by means of directives.

6 Conclusion

A data driven environment based on DD has been presented. The current application uses structured mortar elements, yet completely unstructured environments already have been realized [GEG$^+$95]. Automatized parametric topology generation is rendered possible by an extension language-based mesh generation. The master/slave DS together with the concept of not completely assembled matrices has proved effective for vertex based schemes in DD with matching grid points, i.e. it leads to high modularity and provides identical convergence behavior compared to the non-decomposed domain. The data driven skeleton described has already been used for several applications such as laser optimisation, gyrotron, magnetohydrodynamics, etc. [GEG$^+$95]. The present article is mainly based on the ESP application, which is demanding because of the multidisciplinary nature of the physical problem. The unified data representation together with the common database allows for a modular approach enabling the transparent combination of knowledge in different disciplines and concurrent program development at different sites.

Acknowledgement

This work has been supported by ABB Corporate Research, CH-Baden/Dättwil. We are especially grateful for the opportunity to work on a complex industrially relevant design. The geometry is courtesy of ABB Fläkt, S-35187 Växjö.

REFERENCES

[And87] Anderson D. (1987) Equidistribution Schemes, Poisson Generators, and Adaptive Grids. *Applied Mathematics and Computation* 24: 211–227.

[Bon90] Bonomi E. et al. (1990) Astrid: A Programming Environment for Scientific Applications on Parallel Vector Computers. In Devreese J. and Camp P. V. (eds) *Scientific Computing on Supercomputers II.* Plenum Press, New York.

[CAA87] Connett W. C., Agarwal R., and Achwartz A. L. (1987) An Adaptive Grid-Generation Scheme for Flowfield Calculations. Technical Report AIAA-87-0199, AIAA 25th Aerospace Sciences Meeting, January 12-15 1987, Reno, Nevada.

[Con] Control Data, *ICEM CFD/CAE.*

[DKS80] Dwyer H., Kee R., and Sanders B. (1980) Adaptive Grid Method for Problems in Fluid Mechanics and Heat Transfer. *AIAA Journal* 18(10): 1205–1212.

[EKG97] Egli W., Kogelschatz U., and Gerteisen E. (1997) 3D Computation of Corona, Ion Induced Secondary Flows and Particle Motion in Technical ESP Configurations. 8th Int. Conf. on Electrostatics, 4th-6th June 1997. To appear in the Journal of Electrostatics.

[GEG$^+$95] Gruber R., Egli W., Gerteisen E., Jost G., and Merazzi S. (1995) Problem-Solving Environments: Towards an Environment for Engineering Applications. *SPEEDUP Journal* 9(1).

[Ger94] Gerteisen E. A. (1994) A Generic Data Structure for the Communication of Arbitrary Domain Splitted Mesh Topologies. Technical Report CSCS-TR-94-10.

[Ger96] Gerteisen E. A. (1996) Automatized Generation of Block-Structured Meshes for a Parametric Geometry. Technical Report CSCS/SCSC-TR-96-10.

[MBF$^+$90] Merazzi S., Bonomi E., Flueck M., Gruber R., and Herbin R. (1990) *ASTRID User Manual Rapport GASOV No. 29.* EPFL.

[Mer95] Merazzi S. (1995) *The MEMCOM user manual (version 6.3), B2000 Data Access and Data Description Manual.* SMR Corporation, Bienne, Switzerland.

[Sei86] Seibert W. (1986) An Approach to the Interactive Generation of Blockstructured Volume Grids Using Computer Graphics Devices. In Wesseling P. (ed) *First Int. Conf. on Numerical Grid Generation in CFD*, volume 29, pages 333–342. Landshut, W. Germany, 14th-17th July, 1986.

75

A New Model for the Data Distribution Problem

Thomas Loos and Randall Bramley

1 Introduction

When solving a mesh-discretized PDE on a distributed memory parallel computer, two preliminary problems must be solved: the partitioning of the mesh and the mapping of partition sets to processors. These two define the data distribution problem. All partitioning algorithms try to minimize total computer solution time of the PDE, which is dominated by the execution time of the linear system solver on the resulting matrices. The algorithms attempt to minimize total solution time by approximately minimizing the load imbalance and communications overhead.

Current algorithms model the problem as one of partitioning the graph of the mesh. They estimate load balance in terms of equal size partition sets so that all sub-meshes have nearly the same number of nodes, and communications overhead is measured by the number of edges cut by the partitioning. These criteria are effective for simple iterative methods for solving linear systems, particularly methods based only on matrix-vector products. However, many problems of increasing interest in scientific computing generate linear systems that require preconditioners involving recurrences and other parallelism–inhibiting features.

Using the number of edges cut as a metric ignores the algorithm and data structures used for solving the linear systems. To better estimate the actual cost that partitionings induce on parallel iterative solvers, we define the approximate execution time (AET) of a linear system solver as the sum of its communications, memory, and computational times. The AET metric is calculated using a function that partially and inexpensively simulates the execution of a linear system solver on the target parallel computer.

For iterative linear system solvers the data to be divided are the structurally symmetric sparse matrix A, a sparse preconditioning matrix M, and vectors required to solve the system $Ax = b$. The data distribution problem is that of determining the data layout or distribution among the processors and is usually viewed as an instance of the graph partitioning problem [Saa96].

Graph partitioning algorithms try to minimize the number of edges cut in a graph G by a partitioning P subject to balance conditions. Graph-based metrics for quality

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

other than the number of edges cut have been proposed (see Ashcraft and Liu [AL95] and Rothberg [Rot96] for discussions and comparisons of these metrics). The typical iterative solver user views accuracy and execution time as the most important metrics. The iterative solver execution time ET can be calculated as $ET = \mathrm{NI} \times \mathrm{TPI}$, where NI is the number of solver iterations and TPI is the time per iteration. As a practical matter, NI cannot be determined ahead of time, so only the time per iteration can be estimated. Our results show the edges cut metric does not provide even a qualitative measure of the time per iteration.

A new metric, the approximate execution time (AET) metric, is proposed here to replace the edges cut metric as the cost function to be minimized by a graph partitioning algorithm. It is assumed that the solver algorithm can be stated as a sequence of calls to *kernel* functions. Then, the AET function estimates the time for each kernel function, accounting for the solver input data and computational environment. These kernel estimates are summed to give a time per iteration estimate for the solver; current estimates are within 10% relative error, but are generally much better.

2 The Data Distribution Problem

Classically, the structure of A is viewed as an adjacency matrix $\mathcal{A}$ for the data distribution problem. The data distribution problem is then solved by a graph partitioning algorithm, whose inputs are a graph G, specified in this case by $\mathcal{A}$, the number n of partition sets desired, and potentially an initial *partitioning* or division of G into n partition sets. The graph partitioning algorithm partitions $\mathcal{A}$ into n sets, each each corresponding to one solver process. After the partitioning algorithm is run, a *mapping* algorithm determines the logical process to physical processor mapping. Many graph partitioning algorithms assume G has undirected edges, so $\mathcal{A}$ and by extension, A, are assumed to be structurally symmetric. Frequently P is used to reorder and divide A into block rows or columns; in this paper, we assume that P partitions A into n sets that correspond to n block rows. Also, if process i is assigned a row r of A by P, it is assigned row r of M and of all vectors used by the solver.

Most existing partitioning algorithms minimize the number of edges cut of G by P. This has an apparent mapping to the solver execution time by assuming minimizing the number of edges cut corresponds to minimizing the amount of communication between processes. If it can be further assumed that solver execution time is dominated by communications time, the edges cut metric should predict iterative solver execution time. However, in practice communications time also depends on the actual number of messages since each incurs latency. Simply using the edges cut metric also does not account for the computer network topology, which can affect communications costs, or the solver, preconditioning algorithm, and speed of the parallel processing hardware, all of which affect the numerical costs.

Measuring Solver Execution Time

The iterative solver execution time is initialization time plus $\mathrm{NI} \times \mathrm{TPI}$, where NI is the number of solver iterations, and TPI is the time per iteration. Most of the

Table 1 Numbers of Bi-CGSTAB iterations required Laplace operator matrix, SHERMAN5, and BFS for eight different summation orderings of the dot products. Data from Etsuko Mizukami.

Ordering Method	1	2	3	4	5	6	7	8
Laplace	64	62	63	65	64	60	60	63
SHERMAN5	1034	1025	1024	998	908	974	870	934
BFS	46	49	49	38	37	49	49	38

initialization time is spent in computing a preconditioner, which is typically small compared to total solution time.

It is impossible in general to adequately estimate the number of iterations a preconditioned nonsymmetric iterative solver will take. Although *upper bounds* have been established for the number of conjugate gradient iterations needed for some simple problems with known eigenvalue distributions [AL86], no realistic estimates for practical problems are available. Three additional complicating factors also occur. First, the targeted systems are nonsymmetric. In this case, even a complete *a priori* knowledge of the eigenvalues of the system does not allow estimating the number of iterations. Secondly, all practical solvers use some form of preconditioning. Except in special cases such as diagonally dominant M-matrices, even the existence of the preconditioner is suspect, and its effect on the number of iterations not known. In many cases a preconditioner can actually increase the number of iterations required.

Finally, each domain decomposition implicitly defines a reordering of the matrix with subsequent changes in the order of operations and quality of preconditioning. Table 1 shows the number of iterations required by Bi-CGSTAB for the matrix SHERMAN5 from the Harwell-Boeing collection of test matrices, a steady-state backward-facing step problem in CFD, and the Laplace operator on a $24 \times 24 \times 24$ cube discretized using centered differences. Only the order of summation used in computing the dense dot products was varied; the partitioning and other computations were kept fixed. Even this simple change causes the number of iterations to vary by over 20%. Although this seems an unusual result which indicates an ill-conditioned system or unstable algorithm, it commonly occurs even with well-conditioned problems: the three in Table 1 have estimated condition numbers of 4.2×10^2, 3.6×10^5, and 1.3×10^4. CG-like iterative methods rarely have monotonic convergence with respect to the residual norm, and even CG applied to symmetric positive definite systems characteristically has sharp drops followed by plateaus. If the solver succeeds in reaching the termination residual norm right after a sharp drop, it can take many fewer iterations than if it is just above the termination point. Then the residual norm often stays above the termination level until the next sharp drop is encountered.

Other factors also contribute to the variability of numbers of iterations when the summation order changes. Most modern processors have combined multiply-add units, which send a full 106-bit mantissa from the multiply operation to the add unit. When the dotproduct is distributed across processors, however, the partial sums are rounded to 53-bit IEEE standard mantissas to be sent in a standard 64-bit word, changing the

Table 2 Partitioning Methods Studied

Method	Laplacian EC	BFS EC
Linear	21 364	9 276
Linear-KL	23 542	9 488
Multi-Level	8 792	6 702
Random-KL	22 839	10 674
Scattered-KL	34 532	9 482
Spectral	17 386	7 672
Spectral-KL	14 033	7 684

final sum. Another source of variability is the sensitivity of nonsymmetric iterative methods. Because they typically use an indefinite inner product, large oscillations can occur in the residual norm during the solve. Finally, nonsymmetric linear systems often have poor behaviour not predicted by the spectral condition number. Two well-known examples of this are a large departure from normality and having large magnitude eigenvalues lying close to the imaginary axis.

In a parallel environment, the summation ordering problem is further compounded by the unpredictable order of summation between the processors which affects the matrix-vector as well as the dot product operations. Since determining the number of iterations induced by a domain decomposition is impractical, the AET metric concentrates on the other factor of the total solution time: the time per iteration. This unpredictability of the number of iterations, however, is also unaddressed by the standard edges-cut metric which only targets the communication cost of a single iteration.

The Edges Cut Metric as a Time Per Iteration Predictor

To empirically test the edges cut metric as a predictor of the time per iteration, we used a parallel implementation of van der Vorst's bi-conjugate gradient (Bi-CGSTAB) algorithm [vdV92] with two block preconditioning algorithms: Jacobi/block diagonal (BDIAG) and block SSOR (BSSOR). The preconditioning matrix M consisted of the off-diagonal blocks of A and the factored diagonal blocks of A using incomplete LU factorization with 0 levels of fill (ILU(0)). Other iterative solvers and preconditioners are incorporated in the code, but this combination was chosen as typical of parallel nonsymmetric solvers and uses the kernels found in most parallel iterative methods. The solver was run on an Power Challenge with 2 GB of main memory using 8 of the 10 R-8000 CPUs. Two test matrices were used: the Laplacian operator using seven-point centered differences on a $50 \times 50 \times 60$ domain and *BFS*, a matrix of order 20 284 with 452 752 non-zeroes resulting from solving a refined backward facing step problem. Fig. 1 shows a view of the matrix generated by the Emily [BL94] matrix visualization tool. Seven octa-partitionings of the two matrices were generated using Chaco [HL93] using the methods listed in Table 2. In the table, "KL" means the local Kernighan-Lin [KL70, FM82] method was used as a post-processing step and "EC" is the number of edges cut. The timing results are shown in Fig. 2.

Figure 1 View of the BFS matrix with linear octa-partitioning. The matrix entries appear along the main diagonal. The horizontal and vertical lines represent the partitioning's division of the matrix into blocks.

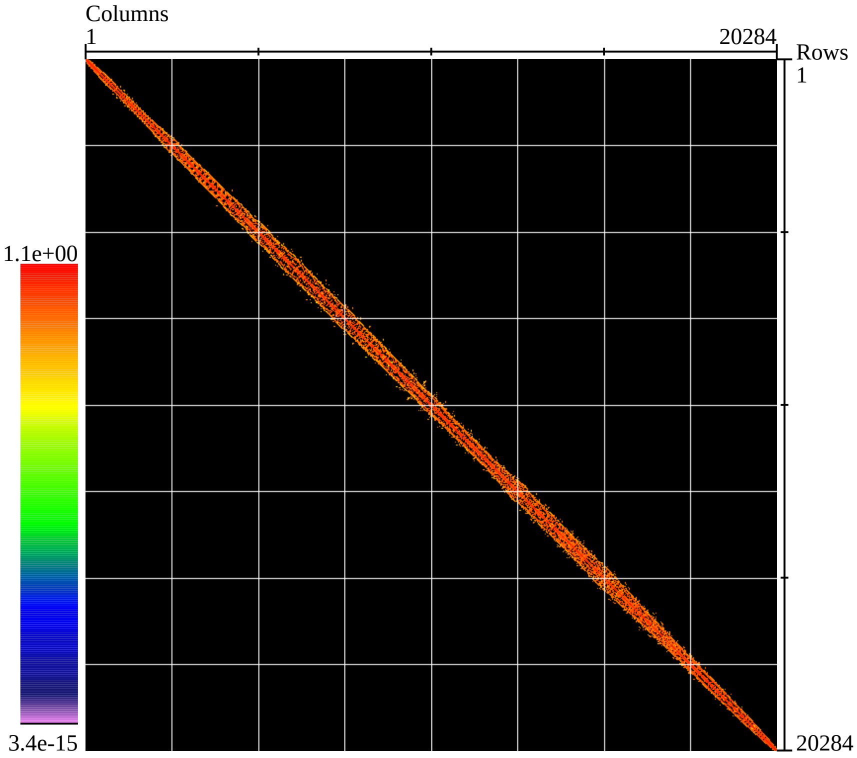
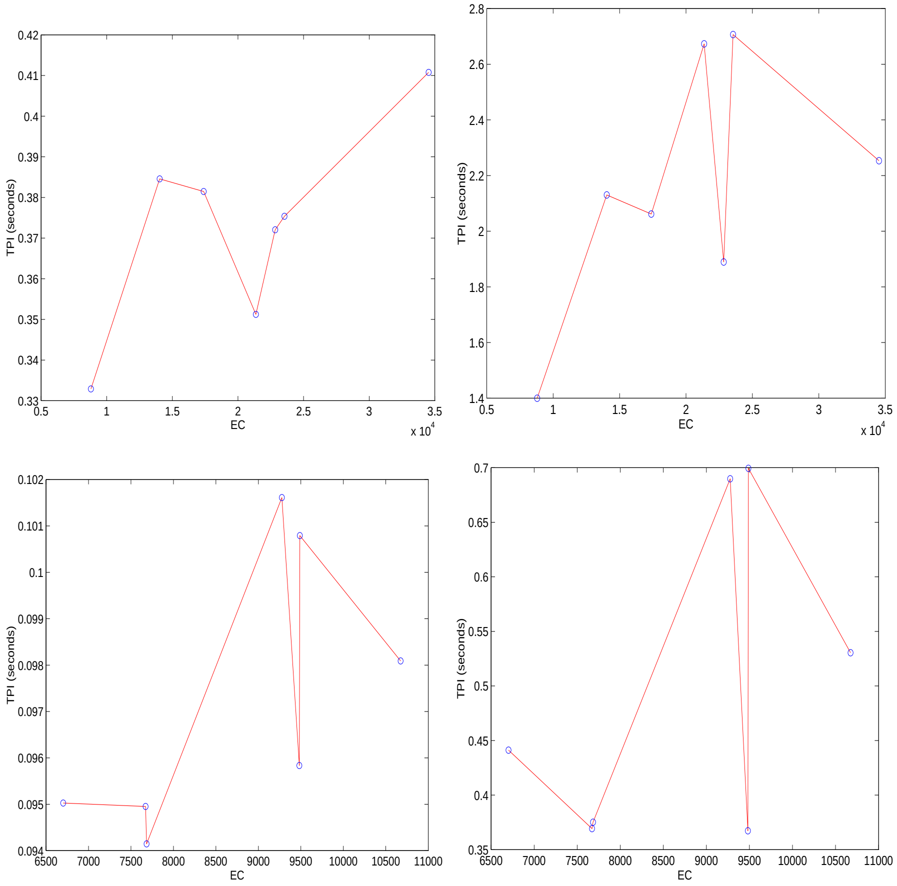


Figure 2 CGSTAB Time Per Iteration results. Top left: Laplacian with BDI AG. Top right: Laplacian with BSSOR. Bottom left: BFS with BDIAG. Bottom right: BFS with BSSOR.



The x-axis of each graph in Fig. 2 shows the number of edges cut for each partitioning method and the y-axis shows the observed time per iteration. If edges cut predicts the time per iteration, each graph in Fig. 2 should be monotone increasing. The simplest matrix/preconditioner pair is the regularly structured Laplacian matrix with the perfectly parallel BDIAG preconditioner. The results for this pair shown in the upper left graph of Fig. 2 indicate edges cut does predict the minimum and and maximum time per iteration correctly, but the function is clearly not monotone increasing. The upper right graph of Fig. 2 shows the results of using the same Laplacian matrix with the BSSOR preconditioner. The number of edges cut does not change, but the time per iteration function is clearly significantly affected, since nothing in the edges cut calculation accounts for a preconditioner change.

The lower left graph of Fig. 2 shows the results of the BFS matrix with BDIAG preconditioning, where the edges cut metric fails to predict the minimum *or* maximum time per iteration. The lower right graph of Fig. 2 shows the results of the BFS matrix using BSSOR preconditioning. The minimum number of edges cut for all partitionings of BFS is 6 702, yet the minimum time per iteration (0.367 s) occurs for the partitioning with 9 488 edges cut, closely followed by the partitionings at 7 672 (0.369 s) and 7 684 EC (0.375 s). The maximum time per iteration (0.699 s) occurs for the partitioning with 9 488 edges cut. The difference in edges cut between the partitionings with the minimum and maximum time per iteration is 6; yet the ratio of maximum to minimum time per iteration is 1.90. The edges cut metric does not even provide a qualitative prediction of the time per iteration.

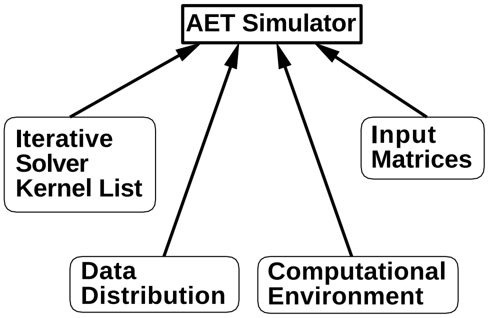
3 The AET Function

Fig. 3 outlines a basic software structure for the approximate execution time (AET) calculation. The AET function is input a list of high-level kernel operations representing the solver algorithm, A and M, low-level kernel timing data for the computational environment(s) used, the data distribution for A, and a representation of the processors in the computational environment. A "building block" approach generates the AET value by simulating a sequence of *kernel* calls.

We assume that a solver can be coded as a sequence of calls to a small number of *kernel* functions. This programming style allows for clear algorithmic statements [BBC^{+}94] and the use of standard numerical and communications library routines, such as the BLAS and the Message Passing Interface standards[MPIF94]. Solvers are expressed in terms of *high-level* kernels; high-level kernels operate on whole matrices and vectors. These high-level kernels are assumed to implemented in terms of either other high-level kernels or *low-level* kernels, which operate on vector and matrix blocks. For the AET calculation, each low-level numerical kernel is timed over a large range of inputs on one processor of the parallel system. Those observations provide a runtime estimation function for the kernel over inputs of arbitrary size. Point-to-point communication and synchronization kernels are timed in a parallel environment [LB96] to build similar models. The key parameters for these low level models for a particular parallel processor are stored in a data file. This allows the view of a parallel computer as a collection of kernel timing models.

The AET function simulates the execution of each high-level kernel in the iterative solver kernel list. Where the high-level kernel calls a low-level kernel, the AET function calculates the low-level kernel's input size and computational environment. From the parallel processor, kernel name, and input size a low-level kernel timing estimate is generated. The high-level kernel estimation routine then combines this estimate with information about cache and synchronization effects and sums the resulting estimate with previous estimates for that kernel to get a high-level kernel execution time estimate. Finally, these high-level kernel estimates are summed to get a time per iteration estimate for each physical processor in the computational environment. For the results below, these time per iteration estimates are averaged over all simulated processors.

Figure 3 AET Software Structure – the rectangles with rounded corners represent data objects. The model allows for changes in the solver, computational environment, and input matrix. This structure is used to generate a fairly accurate estimate of solver runtime.



Kernel Modeling

As an example of low-level kernel modeling, consider the dot product operation. The parallel `dotprod()` high-level kernel can be written in terms of the uniprocessor dot product `ddot()` and parallel reduction `reduce()` low-level kernels as follows:

```
double dotprod(Vector x, Vector y) {
   double sum, answer;
   sum = 0.0;
   for (each block b resident on this processor)
      sum = sum + ddot(x.blocks[b], y.blocks[b]);
   answer = reduce(sum);
   return(sum);  }
```

The AET function supports two low-level kernel modeling methods: a general piecewise linear model and a cached data model. The cached data model estimate is $E(N, S, t_{small}, t_{large}, t_{limit}) = N \cdot t_{est}(N, S, t_{small}, t_{large}, t_{limit})$, where N is the data size, S is the processor cache size S, t_{small} is the time per operation for cache resident data sets, t_{large} is the time per operation for non-cache resident data sets, and t_{limit} is modifies the estimate if the the caching strategy significantly alters the cost per operation. t_{est} is defined by:

$$t_{est} = \begin{cases} t_{small} & , \quad N \leq S \\ \min\{t_{limit}, (St_{small} + (N-S)t_{large})/N\}, & N > S. \end{cases}$$

This approximation for the dot product along with an estimate of the reduction cost [LB96] were summed and used to generate Fig. 4. The left hand graph of the figure shows the total time for dot product operation on one CPU of an SGI Power Challenge. The right hand side graph shows the dot product time per double, which outlines the effect of the t_{small}, t_{large}, and t_{limit} parameters.

Figure 4 Modeling of simple kernels. The left hand graph shows the accuracy of the model and the right hand graph shows the effect of the modeling parameters.

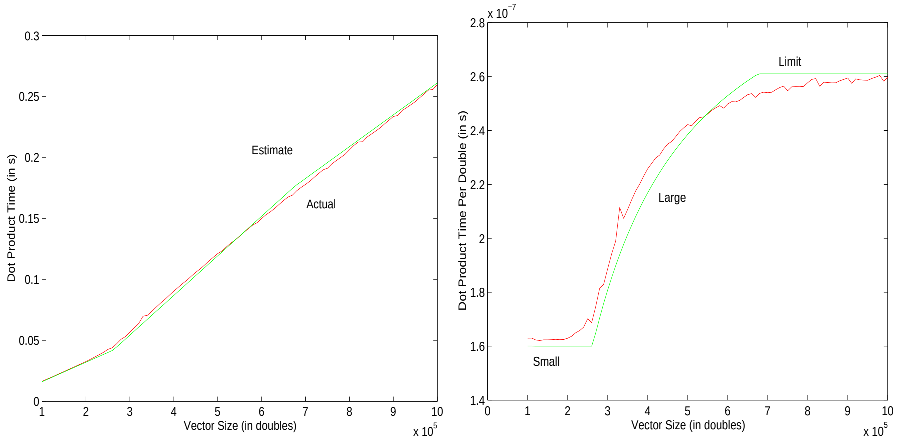


Table 3 AET Results for the Bi-CGSTAB solver using BDIAG preconditioning on the BFS and Lap150 matrices (Laplacian of order 150 000).

# CPUs	Lap150 Act TPI	Lap150 Est TPI	Lap150 Rel Err	BFS Act TPI	BFS Est TPI	BFS Rel Err
1	2.52	2.47	2.14 %	0.789	0.804	1.92 %
2	1.30	1.22	6.38 %	0.372	0.390	4.75 %
4	0.658	0.609	7.52 %	0.174	0.164	5.84 %
8	0.313	0.291	7.10 %	0.0910	0.0824	9.41 %

Complete Modeling

Stand-alone low-level kernel timings are not adequate for approximating the execution time because: (a) the size and distribution of the input matrices A and M are not known *a priori* and (b) residual effects such as the contents of the cache and synchronization delays from previous kernel calls are important. The previous cache contents can and do greatly change the cost of memory accesses [Sto90]; because of the change in cache hit ratios. For example, the vector copy low-level kernel has the ratio $t_{limit}/t_{small} = 8.1$. The AET simulator assumes the use of a least recently used cache replacement policy with a correction constant for other policies such as the random replacement policy used on the SGI Power Challenge. For simplicity, inter-CPU overhead is assumed to be mainly synchronization overhead.

Table 3 shows AET function results on the SGI Power Challenge on two test matrices: the Laplacian operator on a 50 × 50 × 60 domain and *BFS* for the Bi-CGSTAB solver run on 1, 2, 4, and 8 SGI Power Challenge R8000 CPUs using BDIAG preconditioning. Both matrices were partitioned using a linear octa-partitioning. The AET calculation is accurate to within 10 %.

Related Work

Blau's [Bla92] work uses a run-time estimate as input to a partitioning algorithm used by a computer rendering system. This work used previous timing results to predict future timing results; a natural choice for a frame-by-frame renderer, where the input changes a small amount from frame to frame. It did not readily account for changes in the rendering algorithm or computational environment.

Adve [Adv93] and Xu, Zhang, and Sun [XZS96] also use a modeling strategy based on combining empirical observations. They identify segments of a program by first locating communication and synchronization points and computing a *task graph.* Each task then is timed – either in a uniprocessor environment on the same input data in Adve's system or on the same computational environment using a scaled-down version of the input in Xu, Zhang, and Sun's system – and the synchronization and communications routines are separately timed. The task graphs are used to drive a high-level simulator which accounts for inter-process memory contention, communication, and synchronization delays. Our system uses a similar general framework – low-level models based on direct timing observations are combined by a high-level model to get an estimate. As each kernel estimate is independent of the others, task graphs and task timing are not needed once the low-level kernel models are generated. This independence comes at a loss of generality – this model will not work for any but a subset of all parallel programs. However, a more accurate estimate is attainable by focusing on parallel iterative solvers, as this work shows.

4 Conclusions and Future Work

For the data distribution problem, solver run-time is the true metric for a partitioning. In practice the number of solver iterations cannot be predicted, and time per iteration is the primary factor in execution time that can be predicted for iterative linear solvers. We have developed an estimation function that reliably estimates the time per iteration and plan on using it as a cost function for partitioning algorithms. Further work includes using the AET to drive a partitioning algorithm, extending our system to more computational environments, and refining the high-level modeling scheme.

REFERENCES

[Adv93] Adve V. (October 1993) *Analyzing the Behavior and Performance of Parallel Programs.* PhD thesis, University of Wisconsin-Madison.

[AL86] Axelsson O. and Lindskog G. (1986) On the eigenvalue distribution of a class of preconditioning methods. *Numer. Math.* 48: 479–498.

[AL95] Ashrcraft C. and Liu J. (November 1995) Using Domain Decomposition to Find Graph Bisectors. Technical Report CS-95-08, York University.

[BBC$^+$94] Barrett R., Berry M., Chan T., Demmel J., Donato J., Dongarra J., Eijkhout V., Pozo R., Romine C., and van der Vorst H. (1994) *Templates for the Solution of Linear Systems: Buildings Blocks for Iterative Methods.* SIAM, Philadelphia PA, first edition.

[BL94] Bramley R. and Loos T. (July 1994) EMILY: A Visualization Tool for Large Sparse Matrices. Technical Report TR 412A, Indiana University Computer Science

Department.

[Bla92] Blau R. (December 1992) *Performance Evaluation for Computer Image Synthesis Systems*. PhD thesis, University of California - Berkeley. Also available as UCB Techreport CSD-93-736.

[FM82] Fiduccia C. M. and Mattheyses R. M. (1982) A Linear Time Heuristic for Improving Network Partitions. In *Proceedings of the 19th ACM/IEEE Design Automation Conference*, pages 175–181.

[HL93] Hendrickson B. and Leland R. (October 1993) The Chaco User's Guide Version 1.0. Technical Report SAND 93-2339, Sandia National Laboratories.

[KL70] Kernighan B. W. and Lin S. (Feburary 1970) An Efficient Heuristic Procedure for Partitioning Graphs. *Bell Systems Technical Journal* 49: 291–307.

[LB96] Loos T. and Bramley R. (1996) MPI Performance on the SGI Power Challenge. In *Proceedings of the Second MPI Developer's Conference*, pages 203–206. IEEE Computer Society Technical Committee on Distributed Processing.

[MPIF94] Message Passing Interface Forum (May 1994) MPI: A Message-Passing Interface Standard. Technical Report UT-CS-94-230, University of Tennessee, Knoxville.

[Rot96] Rothberg E. (January 1996) Exploring the Tradeoff Between Imbalance and Separator Size in Nested Dissection Ordering. Technical Report none, Silicon Graphics, Inc.

[Saa96] Saad Y. (1996) *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, Boston, MA, first edition.

[Sto90] Stone H. S. (1990) *High Performance Computer Architecture*. Addison-Wesley, Reading, MA, second edition.

[vdV92] van der Vorst H. (1992) Bi–CGSTAB: A fast and smoothly converging variant of Bi–CG for the solution of nonsymmetric linear systems. *SIAM Journal of Scientific and Statistical Computing* 13: 631–644.

[XZS96] Xu Z., Zhang X., and Sun L. (1996) Semi-Emperical Multiprocessor Performance Predictions. Technical Report TR-96-05-01, University of Texas, San Antonio, High Performance Comp. and Software Lab.

76

Parallel Unstructured Mesh Partitioning

C. Walshaw, M. Cross, and M. G. Everett

1 Introduction

The use of unstructured mesh codes on parallel machines can be one of the most efficient ways to solve large Computational Fluid Dynamics (CFD) and Computational Mechanics (CM) problems. Completely general geometries and complex behaviour can readily be modelled and, in principle, the inherent sparsity of many such problems can be exploited to obtain excellent parallel efficiencies. An important consideration, however, is the problem of distributing the mesh across the memory of the machine at runtime so that the computational load is evenly balanced and the amount of interprocessor communication is minimised. It is well known that this problem is NP complete, so in recent years much attention has been focused on developing suitable heuristics, and some powerful methods, many based on a graph corresponding to the communication requirements of the mesh, have been devised, e.g. [FS93]. Closely related to this graph partitioning problem is the problem of optimising existing mesh partitions and in this paper we discuss the partition optimisation problem and its bearing on the graph partitioning problem.

In particular, the algorithms outlined in this paper are designed to address the three problems that arise in partitioning of unstructured finite element and finite volume meshes. Specifically the:

(i) **static partitioning problem** (the classical problem) which arises in trying to distribute a mesh amongst a set of processors;
(ii) **static load-balancing problem** which arises from a mesh that has been generated in parallel;
(iii) **dynamic load-balancing/partitioning problem** which arises from either adaptively refined meshes or meshes in which the computational workload for each cell can vary with time. It can also arise from computing resources with changing patterns of external load (e.g. a network of workstations).

In the latter two cases, (ii) & (iii), the initial data is a distributed graph which may

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

be neither load-balanced nor optimally partitioned. One way of dealing with this is to ship the graph back to some host processor, run a serial static partitioning algorithm on it and redistribute. However, this is unattractive for many reasons. Firstly, an $O(N)$ overhead for the mesh partitioning is simply not scalable if the solver is running at $O(N/P)$. Indeed the graph may not even fit into the memory of the host machine and may thus incur enormous delays through memory paging. In addition, a partition of the graph (which may even be optimal) already exists, so it makes sense to reuse this as a starting point for repartitioning [WB95]. In fact, not only is the load-balancing likely to be unnecessarily computationally expensive if it fails to use this information, but also the mesh elements will be redistributed without any reference to their previous 'home processor' and heavy data migration may result. Thus, because the graph is already distributed, it is a natural strategy to repartition it *in situ*.

The algorithms developed here are therefore designed to iteratively optimise and if necessary load-balance an existing partition in parallel. In the first case, (i) above, an initial partition is generated using a fast but suboptimal partitioner such as the greedy algorithm and then the data is distributed.

2 Optimisation

In this section we present two complementary iterative algorithms which combined together form a powerful and flexible technique for optimising unstructured mesh partitions. Initially the subdomain heuristic attempts to 'improve' the 'shape' of the subdomains. However, this heuristic cannot guarantee load-balance and so a second heuristic, a parallel version of the Kernighan-Lin algorithm, [KL70], which also incorporates load-balancing is applied to share the workload equally between all subdomains and to carry out local refinement.

The subdomain heuristic

The idea behind the subdomain heuristic is to minimise the surface energy of the subdomains (in some graph sense). This is achieved by each processor determining the centre of its subdomain and then measuring the radial distance from the centre to the border of the subdomain and attempting to minimise this by migrating vertices which are furthest away.

Determining the 'centre' of a subdomain is relatively easy and can be achieved by moving in level sets inwards from the subdomain border until all the vertices in the subdomain have been visited. The final set defines the centre of the subdomain and, if the graph is connected (assumed), the level sets will completely cover the subdomain, although the centre may not be a connected set of vertices. The reverse of this process can then be used to determine the radial distance.

Having derived these sets each vertex can be marked by its radial distance. Nodes that are not connected to the centre are not marked and this is useful for migrating small disconnected parts of a subdomain to more appropriate processors. Neighbouring processors are informed of the radial distances of vertices on their borders and the vertices are migrated according to a combination of load-imbalance, radial distance and the change in cut-edges. This decision process is fully described in [WCE95a].

Local refinement & load-balancing

Having achieved approximate load-balance and good global subdomain shapes, a process of local refinement and exact load-balancing takes place.

The gain and preference functions. A key concept in the method is the idea of gain and preference functions. Loosely, the gain $g(v, q)$ of a vertex v in subdomain S_p can be calculated for every other subdomain, S_q, $q \neq p$, and expresses some 'estimate' of how much the partition would be 'improved' were v to migrate to S_q. The preference $f(v)$ is then just the value of q which maximises the gain – i.e. $f(v) = q$ where q attains $\max_{r \in P} g(v, r)$.

The gain is usually directly related to some cost function which measures the quality of the partition and which we aim to minimise. Typically the cost function used is simply the total weight of cut edges, $|E_c|$, and then the gain expresses the change in $|E_c|$. More recently, however, there has been some debate about the most important quantity to minimise and in [VK95], Vanderstraeten *et al.* demonstrate that it can be extremely effective to vary the cost function based on a knowledge of the solver. Meanwhile, in [WCE$^+$95c] we show that the architecture of the parallel machine and how the partition is mapped down onto its communications network can also play an important role. Whichever cost function is chosen, however, the idea of gains is generic. For the purposes of this paper, however, we shall assume that the gain $g(v, q)$ just expresses the reduction in the cut-edge weight, $|E_c|$.

Load-balancing. The load-balancing problem, i.e. how to distribute N tasks over a network of P processors so that none have more than $\lceil N/P \rceil$, is a very important area for research in its own right with a vast range of applications. Here we use an elegant technique recently developed by Hu & Blake, [HB95], related to, but with faster convergence than the commonly used diffusive methods, e.g. [GMS95], and which minimises the Euclidean norm of the transferred weight.

This algorithm (or, in principle, any other distributed load-balancing algorithm) defines how much weight to transfer across edges of the subdomain graph and we then use the local refinement mechanism to decide which vertices to move.

The parallel local refinement mechanism. An algorithm which comes to mind for local refinement purposes is the Kernighan-Lin (KL) heuristic, [KL70], and in particular a linear-time variant proposed by Fiduccia & Mattheyses (FM), [FM82]. We use an algorithm largely inspired by the KL/FM algorithms but with several modifications to better suit our purposes. In particular, only boundary vertices are allowed to migrate and only to neighbouring processors.

The algorithm, which is fully described in [WCE97b, WCE95b], is thus run in the boundary regions of the subdomains and at each iteration a processor, p, calculates the preference and gain of its own border vertices and the desired flow across each p-q interface with neighbouring processors q and a halo update is carried out. Next, for each interface, the processor to which it has been assigned, p say, creates a bucket list structure (as in the FM algorithm) for border vertices v owned by itself which have preference $f(v) = q$ and halo vertices u owned by q which have preference $f(u) = p$. Vertices are then iteratively selected from either subdomain so as to firstly satisfy the flow as far as possible and secondly maximise the gain as much as possible.

3 Graph Reduction

For coarse granularity partitions it is inefficient to apply the optimisation techniques to every graph vertex as most will be internal to the subdomains. A simple technique to speed up the optimisation process, therefore, is to group vertices together to form *clusters*, use the clusters to define a new graph, recursively iterate this procedure until the graph size falls below some threshold and then apply the partitioning algorithm to these reduced size graphs. This is quite a common technique and has been used by several authors in various ways – for example, in a multilevel way analogous to multigrid techniques [BS94, HL95], and in an adaptive way analogous to dynamic refinement techniques, [WB95].

Reduction

To create a coarser graph $G'(V',E')$ from $G(V,E)$ we use a variant of the edge contraction algorithm proposed by Hendrickson & Leland, [HL95]. The idea is to find a maximal independent subset of graph edges and then collapse them. The set is independent because no two edges in the set are incident on the same vertex (so no two edges in the set are adjacent), and maximal because no more edges can be added to the set without breaking the independence criterion. Having found such a set, each selected edge is collapsed and the vertices, $u_1, u_2 \in V$ say, at either end of it are merged to form a new vertex $v \in V'$ with weight $|v| = |u_1| + |u_2|$. Edges which have not been collapsed are inherited by the reduced graph and, where they become duplicated, are merged with their weight summed. This occurs if, for example, the edges (u_1, u_3) and (u_2, u_3) exist when edge (u_1, u_2) is collapsed. Because of the inheritance properties of this algorithm, it is easy to see that the total graph weight remains the same, $|V| = |V'|$. The total edge weight is reduced (by an amount equal to the weight of the collapsed edges), but the weight of the cut edges remains the same, $|E_c| = |E'_c|$.

Parallel matching

A simple way to construct a maximal independent subset of edges is to visit the vertices of the graph in a random order and pair up or match unmatched vertices with a random unmatched neighbour. For the parallel version we use more or less the same procedure; each processor visiting in parallel the vertices that it owns. We modify the matching algorithm, however, by always matching with a local vertex in preference to a vertex owned by another processor. The local matching can take place entirely in parallel but usually leaves a few boundary vertices who have no unmatched local neighbours but possibly some unmatched non-local neighbours.

The simplest solution would be to terminate the matching at this point. However, in the worst-case scenario if the initial partition is particularly bad and most vertices have no local neighbours (for example a random partition), little or no matching may have taken place. We therefore continue the matching with an parallel iterative procedure which finishes only when there are no vertices unmatched. Nodes which are matched across interprocessor boundaries are migrated to one of the two owning processors and then the construction of the reduced graph can take place entirely in parallel.

The algorithm is fully described in [WCE97b].

4 Results

The software tool written at Greenwich to implement the optimisation techniques is known as JOSTLE. This software has been previously demonstrated to provide partitions of higher quality than MRSB, [WCE95a], and here, due to space constraints, we concentrate on parallel timings and the effect of the initial partition. Results which demonstrate the optimisation techniques applied to adaptively refined meshes can be found in [WCE97a] and the use of JOSTLE for mapping partitions onto machine topologies can be found in [WCE$^+$95c].

Metrics

We use two metrics to measure the performance of the algorithms, the total weight of cut edges, $|E_c|$ and $t(s)$, the execution time in seconds. The best measurement of the partition quality, and ultimately the only important one, is the parallel efficiency of the application from which the graph arises on a given machine. Unfortunately, however, this efficiency will depend on many things – typically the machine (size, architecture, latency, bandwidth and flop rate), the solution algorithm (explicit, implicit with direct linear solution, implicit with iterative linear solution) and the problem itself (size, no. of iterations) all play a part (see also Section 2). As a result it is impossible to fully assess a partitioning method independent of the solver and the machine to be employed and to do so goes beyond the scope of this paper. Here, therefore, we use $|E_c|$ to give a rough indication of the volume of communication traffic.

Parallel timings

Achieving high parallel performance for parallel partitioning codes such as JOSTLE is not as easy as, say, a typical CFD or CM code. For a start the algorithms use only integer operations and so there are no MFlops to 'hide behind'. In addition, most of the work is carried out on the subdomain boundaries so very little of the actual graph is used. Also the partitioner itself may not necessarily be well load-balanced and the communications cost may dominate on the coarsest reduced graphs. On the other hand, as was explained in Section 1, partitioning on the host may be impossible or at least much more expensive and if the cost of partitioning is regarded (as it should be) as a parallel overhead, it usually extremely inexpensive relative to the overall solution time of the problem.

Tables 1 and 2 give some typical results for 2D (`tri60k`) and 3D (`brack2`) meshes on the Edinburgh Cray T3D with up to 128 processors. These demonstrate very good speedups for this sort of code and more importantly, very low overheads (of the order of a few seconds) for the parallel partitioning. Note that the $|E_c|$ results obtained for the parallel version of JOSTLE may not be exactly the same as those of the serial version, due to different orderings of linked lists, but that, since these are random orderings, there are no consistent differences in quality.

Table 1 Results for tri60k mesh: $N = 60005$, $E = 89440$

	serial		parallel		
P	$\lvert E_c\rvert$	$t(s)$	$\lvert E_c\rvert$	$t(s)$	speed up
16	1104	14.64	1093	2.65	5.52
32	1669	15.67	1668	1.88	8.33
64	2530	19.79	2572	1.66	11.92
128	3698	24.32	3721	1.29	18.85

Table 2 Results for brack2 mesh: $N = 62032$, $E = 121544$

	serial		parallel		
P	$\lvert E_c\rvert$	$t(s)$	$\lvert E_c\rvert$	$t(s)$	speed up
16	13717	35.19	13442	7.13	4.93
32	21098	40.85	21004	4.90	8.34
64	30407	54.21	30276	4.33	12.52
128	43109	59.40	42959	2.65	22.41

The initial partition

We have tested the optimisation techniques with a variety of initial partitioning algorithms. Two crude techniques are *random* partitioning which assigns the vertices randomly and *block* partitioning which assigns the first N/P vertices to processor 0, the next N/P to processor 1, etc. These are attractive as the data can, in principle, be input in parallel. However, random partitioning gives something close to a worst-case initial partition and block partitioning can be very poor, particularly in the case of an advancing front mesh generator (as used for the whitaker3 mesh) where the mesh elements spiral in towards the centre. A slightly more effective technique is geometric sorting where the elements are sorted according to their x, y (and z for 3D) coordinates and each dimension is partitioned in a strip-wise fashion. This too can be carried out in parallel (using a parallel sorting algorithm) but can create long thin and sometimes multiply connected domains. The final algorithm we have tested is the Greedy algorithm [Far88]. This is clearly seen to be the fastest *graph-based* method as it only visits each graph edge once, but can only be applied in serial.

Table 3 Different initial partitions for whitaker3: $N = 9800$, $E = 28989$, $P = 32$

initial algorithm	initial $\lvert E_c\rvert$	optimised $\lvert E_c\rvert$	$t(s)$
random	28083	1848	3.61
loop	11571	1882	2.02
geosort	1854	1818	2.26
greedy	2143	1805	2.12

Tables 3 & 4 show the results obtained from a Sun 20 with a 75 MHz CPU and 128 Mbytes of memory. As can be seen, the quality of the final optimised partition does

Table 4 Different initial partitions for `barth5`: $N = 15606$, $E = 45878$, $P = 64$

initial algorithm	initial $\|E_c\|$	optimised $\|E_c\|$	$t(s)$
random	45174	3096	7.38
loop	10643	2930	4.04
geosort	5416	2905	4.20
greedy	4046	2970	3.88

not vary significantly with the initial partitioner chosen (except for a little noise) and thus the optimisation techniques are demonstrated to be very powerful, in particular for the worst-case *random* partition. What is affected however is the partitioning time and in general, as might be expected, the poorer the quality of the initial partition, the longer it takes to optimise it.

5 Conclusion

We have outlined a new method for optimising graph partitions with a specific focus on its application to the mapping of unstructured meshes onto parallel computers. In this context the static graph-partitioning task can be very efficiently addressed through a two-stage procedure – one to yield a legal initial partition and the second to improve its quality with respect to interprocessor communication and load-balance. The method is further enhanced through the use of a clustering technique. For the experiments reported in this paper the cost of parallel partitioning is shown to be of the order of a few seconds even for relatively large graphs. In addition, the partition quality is shown to be reasonably independent of the initial partition.

Acknowledgement

We would like thank the Edinburgh Parallel Computer Centre and the Engineering and Physical Sciences Research Council for access to the Cray T3D.

REFERENCES

[BS94] Barnard S. T. and Simon H. D. (1994) A Fast Multilevel Implementation of Recursive Spectral Bisection for Partitioning Unstructured Problems. *Concurrency: Practice & Experience* 6(2): 101–117.

[Far88] Farhat C. (1988) A Simple and Efficient Automatic FEM Domain Decomposer. *Comp. & Struct.* 28(5): 579–602.

[FM82] Fiduccia C. M. and Mattheyses R. M. (1982) A Linear Time Heuristic for Improving Network Partitions. In *Proc. 19th IEEE Design Automation Conf.*, pages 175–181. IEEE, Piscataway, NJ.

[FS93] Farhat C. and Simon H. D. (1993) TOP/DOMDEC – a Software Tool for Mesh Partitioning and Parallel Processing. Tech. Rep. RNR-93-011, NASA Ames, Moffat Field, CA.

[GMS95] Ghosh B., Muthukrishnan S., and Schultz M. H. (1995) Faster Schedules for Diffusive Load Balancing via Over-Relaxation. TR 1065, Department of Computer Science, Yale University, New Haven, CT 06520, USA.

[HB95] Hu Y. F. and Blake R. J. (1995) An optimal dynamic load balancing algorithm. Preprint DL-P-95-011, Daresbury Laboratory, Warrington, WA4 4AD, UK.

[HL95] Hendrickson B. and Leland R. (1995) A Multilevel Algorithm for Partitioning Graphs. In *Proc. Supercomputing '95.*

[KL70] Kernighan B. W. and Lin S. (February 1970) An Efficient Heuristic for Partitioning Graphs. *Bell Systems Tech. J.* 49: 291–308.

[VK95] Vanderstraeten D. and Keunings R. (1995) Optimized Partitioning of Unstructured Computational Grids. *Int. J. Num. Meth. Engng.* 38: 433–450.

[WB95] Walshaw C. H. and Berzins M. (1995) Dynamic load-balancing for PDE solvers on adaptive unstructured meshes. *Concurrency: Practice & Experience* 7(1): 17–28.

[WCE95a] Walshaw C., Cross M., and Everett M. (1995) A Localised Algorithm for Optimising Unstructured Mesh Partitions. *Int. J. Supercomput. Applics.* 9(4): 280–295.

[WCE95b] Walshaw C., Cross M., and Everett M. (1995) Dynamic mesh partitioning: a unified optimisation and load-balancing algorithm. Tech. Rep. 95/IM/06, University of Greenwich, London SE18 6PF, UK.

[WCE$^+$95c] Walshaw C., Cross M., Everett M., Johnson S., and McManus K. (1995) Partitioning & Mapping of Unstructured Meshes to Parallel Machine Topologies. In Ferreira A. and Rolim J. (eds) *Proc. Irregular '95: Parallel Algorithms for Irregularly Structured Problems*, volume 980 of *LNCS*, pages 121–126. Springer.

[WCE97a] Walshaw C., Cross M., and Everett M. (1997) Dynamic load-balancing for parallel adaptive unstructured meshes. In Heath *et al* M. (ed) *Parallel Processing for Scientific Computing.* SIAM, Philadelphia.

[WCE97b] Walshaw C., Cross M., and Everett M. (1997) Parallel Unstructured Mesh Partitioning. (in preparation).

Domain Decomposition and Multilevel Methods in Diffpack

Are Magnus Bruaset, Hans Petter Langtangen, and

Gerhard W. Zumbusch

1 Introduction

Looking back a decade or two, the computing power commonly available to scientists and engineers has grown at an amazing rate. Following this race for megaflops, the scientific community has developed a taste for solving mathematical problems of increasing levels of complexity. Naturally, this trend calls for sophisticated numerical methods that are capable of solving the problems in question in an efficient, yet reliable, way.

Since the bottleneck of many scientific applications turns out to be the numerical solution of linear or nonlinear systems of equations, this field has been subject to intensive research over the years. In particular, much attention has been paid to domain decomposition and multigrid methods, which have proven to be highly efficient strategies for many types of applications. Although the performance of such methods has been theoretically analyzed, extensive numerical experimentation is usually required in more complicated applications in order to obtain the best possible results. From this point of view, there is a need for software environments with genuine support for these types of numerical experiments, giving the user access to different algorithmic scenarios at the press of a button.

Domain decomposition and multilevel methods contain a variety of more standard numerical building blocks (linear solvers, matrix assembly, interpolation of fields, etc.). Successful software for complicated applications must offer the user a flexible run-time combination of all these different components. The purpose of the present paper is to describe how one can achieve such flexible software. In particular, we present a unified framework for domain decomposition and multilevel methods, and show how this framework can be efficiently implemented in existing software packages for PDEs[1].

The unified framework about to be presented is in part well known from the analysis

[1] The software design discussed in this paper has been implemented and verified using the object-oriented PDE library Diffpack [Dif, BL97].

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

of overlapping and non-overlapping methods [DW90], as well as from theory for overlapping and multilevel schemes [Xu92]. In this context, the goal of this paper is to extend the known framework to cover even more methods in common use, especially some Schur complement and nonlinear schemes. We will formulate the framework in a novel way that encourages systematic implementation of a wide class of domain decomposition and multilevel methods. Finally, we report on the experiences gathered from a particular implementation in the Diffpack software.

2 The Unified Framework

Abstract Schwarz Method

We consider linear systems, $Ax = f$, where A arises from the discretization of a partial differential operator. The prototype solution algorithm, the additive Schwarz method, can be written as

$$B \;=\; \sum_j R_j^* \, B_j \, R_j \tag{2.1}$$

with approximate solvers B_j operating on a subspace V_j and restrictions R_j and adjoint interpolations R_j^*. In particular we consider the following methods and its variants (see also [SBG96]), which all can be written in the framework:

multilevel iteration	additive, multiplicative, nonlinear
overlapping Schwarz	additive, multiplicative, nonlinear with, without coarse grid
Schur complement iteration	exact, inexact local solves
Schur complement preconditioner	Neumann-Neumann, wirebasket with, without coarse grid

The interface basically consists of

- transfer or restriction operators such as R_j and R_j^*,
- (approximate) subspace solvers B_j,
- evaluation of residuals $(f - A_j x)$ on a subdomain (optional).

Here, A_j is the discrete operator on V_j. The appropriate solver B_j is normally chosen as a traditional linear or nonlinear method, whereas the "transfer" R_j is usually implemented via sparse matrices or difference stencils. However, also algorithmic implementations of R_j and message passing in a parallel environment are possible. Software components for B_j, R_j, R_j^* and $f - A_j x$ are normally found in a package for PDEs.

Multilevel Methods

Standard multigrid algorithms fit into the frame outlined in 2. In the linear multigrid context the abstraction of the (not necessarily adjoint) grid transfer R_j, R_j^* and the local pre- and post-smoothers B_j, B_j^* is sometimes referred to as abstract multigrid.

The implementation follows equation (4.1.1) in Hackbusch [Hac85]. Using multigrid as a preconditioner implies the initial guess $x = 0$.

However, it is also possible to include nonlinear multigrid with nonlinear smoothers B_j, B_j^* [Zum96a]. Using equation (9.3.3) in [Hac85], we can implement the nonlinear FAS scheme and the nested nonlinear multigrid version by Hackbusch. The pre- and post-smoothers are now nonlinear iterative solvers.

This abstract multilevel approach can be used for different variants of multigrid. The particular algorithm depends on the initialization and implementation of the smoothers B_j, the transfer operators R_j, and the operators A_j. There may be non-nested, non-matching, or adaptively refined grids, operators defined by Galerkin products or operator dependent transfers, or algebraic multigrid transfers and operators.

Schur Complement Methods

We decompose the stiffness matrix into one part related to the coupling interface c and several independent parts related to the interior nodes of the subdomains $j = 1, \dots, m$.

$$A = \left(\begin{array}{cccc|c} A_{11} & & & & A_{1c} \\ & A_{22} & & & A_{2c} \\ & & & \ddots & \vdots \\ \hline A_{c1} & A_{c2} & & \dots & A_{cc} \end{array}\right)$$

The Schur complement is defined by $S = A_{cc} - \sum_j A_{cj} A_{jj}^{-1} A_{jc}$.

In the case we solve subdomain problems exactly or accurate enough, or we are able to compute S itself rather than the action of S, the equation system reduces to a system $S\, x_c = f_c$ for the unknowns on the interface x_c.

Neumann-Neumann Preconditioner for the Schur Complement. Preconditioning of the interface system with a Neumann-Neumann algorithm [BGLV89] looks like equation 2.1 in the framework of additive Schwarz methods. The operators B_j now solve homogeneous Neumann problems obtained by sub-assembly on a subdomain. The transfer operators R_j and R_j^* copy (scaled) nodal data from the interface x_c to the nodes on the boundary of the subdomain and vice versa. Hence the ordinary additive Schwarz implementation can be used.

A coarse grid can be added to the Neumann-Neumann preconditioner without changes of the algorithm [DW95] just like in the overlapping Schwarz case adding one subspace $j = 0$. Coupling a coarse grid equation in a multiplicative symmetric way instead, called balancing [Man93], requires some extensions resulting in a mixture of the additive and the multiplicative Schwarz algorithm. One can use the standard global-to-local communication pattern and additional residual evaluations to create the multiplicative coupling.

Wire Basket Preconditioner for the Schur Complement. A preconditioner of wirebasket type [Smi91] for the Schur complement may also be written as an additive Schwarz method now for subspaces of the interface consisting of vertices v, single faces f, and single edges e,

$$B_c = R^{v*} B^v R^v + \sum_j R_j^{e*} B_j^e R_j^e + \sum_k R_k^{f*} B_k^f R_k^f$$

The transfer operators R_j^e and R_k^f perform a hierarchical basis transform. The vertex solver B^v can be implemented by a standard coarse grid solver and the edge solvers B_j^e can be substituted by diagonal scaling. However, the face solvers B_k^f have to be implemented as preconditioners for an interface problem covering both adjacent subdomains.

The two-dimensional analog BPS has a similar structure. It can be implemented by leaving out the faces and choosing some local interface preconditioner for the edge solvers B_j^e [BPS86].

Inexact Schur Complement. Introducing approximate local Dirichlet solvers B_j in the computation of the Schur complement $S \approx A_{cc} - \sum_j A_{cj} B_j A_{jc}$, one cannot restrict the computations to $Sx_c = f_c$, but the full system $Ax = f$ must be considered. We seek a preconditioner for the matrix A, which is constructed using a preconditioner B_c for S and preconditioners $\bar{B}_j$ for the local Dirichlet type problems A_{jj} that may differ from B_j used for the Schur complement. However, the inexact solvers B_j in the Schur complement method are still not fully understood theoretically [HLM91].

We write the Schur decomposition in the additive Schwarz framework

$$B \;=\; R_c^* \, B_c \, R_c \;+\; \sum_j R_j^* \, \bar{B}_j \, R_j$$

with restrictions $R_j x = x_j$ and the transformation to the Schur system $R_c x = x_c - \sum_j A_{cj} B_j x_j$.

One can use multilevel and domain decomposition methods or standard iterative solvers for the implementation of the local solvers B_j, B_j^*, $\bar{B}_j$, and the local solvers used in B_c. This approach to the Schur complement can also be used for nonlinear problems in a Picard iteration manner (see [Zum96b]) since the residual for the full system has to be evaluated every outer iteration step anyway. An exact Dirichlet solver B_j leads to zero residuals at interior nodes and is more efficiently implemented computing on the interface directly as in the previous section.

3 Realizing the Framework

As mentioned initially, the abstract view of the different domain decomposition and multilevel methods taken in the description above has been realized in terms of software. More precisely, the described framework is the basis for the implementation of domain decomposition and multilevel algorithms in the Diffpack simulation environment. Diffpack [Dif] consists of a set of object-oriented libraries written in C++, intended to simplify the implementation of PDE solvers [BL97]. The involved libraries contain many useful abstractions that come quite natural to developers of PDE software. For instance, the application programmer has immediate access to high-level abstractions for linear systems, various matrix formats, different solvers and preconditioners, as well as building blocks for finite element and finite difference discretizations.

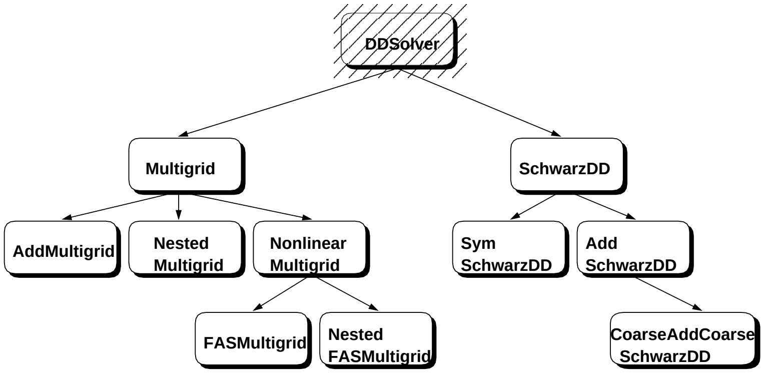
Originally, Diffpack was designed without attention to domain decomposition or multilevel algorithms. Nevertheless, by adopting a unified approach to such methods, one can in principle implement this functionality as an add-on module to existing PDE packages. However, it is then crucial that the framework organizing the different methods can take advantage of tools already present in the available software platform. During the course of the multilevel extension of Diffpack, we have experienced that a clean design of such an add-on module depends heavily on a clean design and modular structure of the underlying libraries. We believe that abstract data types and object-oriented programming are important mechanisms for achieving the necessary degree of modularity. In fact, the Diffpack code for the described framework was realized as a high-level, compact combination of existing C++ classes. We refer to [ABL97, BL97, ABC$^+$97, BL96] and the references therein for information about object-oriented numerics, the efficiency of C++ for scientific computing, the design of Diffpack and examples on Diffpack applications. It should be mentioned that object-oriented implementations of domain decomposition and multilevel strategies have also been addressed by other authors, e.g. as part of the PETSc system [GS94, PET].

Equation solvers in Diffpack utilize abstractions for linear systems, as well as linear and nonlinear solvers. In this context, a linear system consists of a coefficient matrix, a solution vector, a right-hand side and possibly a preconditioner, which will be reused in our framework. The preconditioner can either be a matrix or an *action.* In case of an action, the preconditioner can, e.g., call a linear solver for the same or a related PDE problem. Linear solvers can utilize convergence monitors in order to control the degree of solution accuracy [BL96]. At its present stage of development, Diffpack offers relatively simple nonlinear solvers, like Newton's method or the Picard iteration. This type of algorithm requires the programmer to define and solve a linear (sub)system. Operators can be defined in terms of coefficient matrices arising from finite element assembly. Since a finite element grid is just a C++ object in Diffpack, it is easy to create a hierarchy of grids, and apply toolboxes for finite element schemes and the PDE's definition to create the associated operators and right-hand sides.

The layered, modular design of Diffpack building blocks can be immediately applied to create linear and nonlinear operators, smoothers, transfer operators, residuals and other basic components needed in domain decomposition and multilevel methods, reusing existing code. For instance, the subspace solver B_j in a domain decomposition method may be implemented as a linear solver call or a nonlinear solution procedure, both present in the original package. To allow maximum flexibility, the programmer of the PDE application is responsible for defining B_j, and contrary to the PETSc approach [GS94, PET], we do not *require* a reference to a linear solver object. The main ingredients of the domain decomposition interface are the procedures for grid transfer and for local solvers. Data is only stored for some auxiliary vectors and parameters like the multigrid cycle type (V, W, ...).

The technical details of taking an existing Diffpack application and equipping it with domain decomposition and multilevel methods are described elsewhere [Zum96a, Zum96b]. However, due to the flexibility of the original software components, it turns out to be trivial to run a multigrid solver and experiment with various pre- and post-smoothers (choice of algorithm, number of sweeps, order of unknowns), coarse grid solvers (iterative and direct, grid size), cycle-types, nested iterations, non-matching grids, semi-coarsening, multigrid used as a preconditioner or as a stand-alone solver, different nonlinear versions, grid types and special procedures to initialize operators. For domain decomposition, the type, precision and termination of subdomain solvers, the decomposition of the domain, the type of a coarse grid and coarse-grid solver, and the scaling of transfer operators are of main

Figure 1 Multilevel and Domain Decomposition algorithms. Abstract `DDSolver`, the multiplicative and additive multigrid algorithms including nonlinear versions and additive, multiplicative, symmetric multiplicative and mixed additive/ multiplicative Schwarz algorithms.



interest. Consequently, the resulting software environment satisfies the most important requirement stated in the introduction of this paper; to offer the user genuine support for systematic numerical experiments with sophisticated multilevel strategies.

As previously mentioned, Schur complement methods do not immediately fit into a unified framework. This is also reflected in the pilot implementation. We use an implicit representation of the Schur complement, implemented as an algorithmically defined matrix. This is basically a new type of matrix in Diffpack, realized as a subclass in the existing matrix hierarchy [BL96], which implements the matrix product Sx only. New and old application software can of course work with this matrix type through an abstract (base class) interface. The action Sx is implemented by calling (reusing) subdomain Dirichlet solvers for A_{jj}^{-1} and several matrix multiplications, since the concept of a Schur complement was not present in Diffpack. Direct access to S is then not available. However, standard Schwarz preconditioners can be used for the Schur complement system, reusing even the domain decomposition algorithms.

The domain decomposition and multilevel methods introduced in Diffpack are organized in a class hierarchy with `DDSolver` as base class, see Figure 1. Various specific solution strategies are organized as subclasses of either (multiplicative) `Multigrid` or (alternating) Schwarz domain decomposition (`SchwarzDD`). `Multigrid` uses level to level data transfer ($j \to j+1$), while the `SchwarzDD` algorithms uses local to global data transfer. `SymSchwarzDD` is a symmetric version of the multiplicative `SchwarzDD`, cycling back and forth like the multiplicative `Multigrid`. The `CoarseAddCoarseSchwarzDD` version is used for special treatment of the coarse grid, while the subdomains are treated as in the additive `AddSchwarzDD`. The hierarchy itself is designed to optimize code reuse and reflects software issues, while one can certainly think of modifications in the categorization. The subclasses make use of existing solvers and preconditioners in Diffpack, while still offering the duality of being accessible as new solvers and new preconditioners in the original libraries. For the authors, this experience of playing around with abstractions and extending libraries in ways that were not initially planned for, has been a strong indication that modern programming techniques, such as object-oriented programming, are vital for an accelerated development of scientific computing.

4 Efficiency

We have outlined a flexible software framework for domain decomposition and multilevel methods, where the particular Diffpack implementation is in C++. Many will expect the computational efficiency of such flexible implementations and the use of C++ to be significantly worse than special-purpose Fortran codes tailored at a specific PDE and solution algorithm. To shed some light on this problem we have performed some simple numerical experiments with multigrid methods for a Poisson equation, with smooth variable coefficients, on the unit square. A general Diffpack implementation, also applicable to unstructured grids, was compared to (a) the adaptive PLTMG code [Ban94], (b) a finite difference based example Fortran code with MPI [Dou95a], and (c) a constant 5-point stencil sparse matrix Fortran Poisson solver Madpack5 [Dou95b]. The table below shows CPU times for various problem sizes.

level j	4	5	6	7	8	9
size n	289	1089	4225	16641	66049	263169
PLTMG	.43	1.7	5.7	30	140	–
MPI example	.15	.17	.23	.49	1.8	7.6
madpack5	.01	.04	.12	.50	2.8	13
Diffpack	.07	.16	.39	1.5	7.1	24

As we see, there is no indication that the object-oriented implementation style in C++ implies a significant loss of computational efficiency. The reasons for this are simple; object-orientation is only used for high-level administration in Diffpack, whereas CPU-time consuming operations usually take place in low level C/Fortran-style routines that can be highly optimized by today's compiler technology. Moreover, the general finite element software in Diffpack makes use of simplified, optimized algorithms when it is known that the grid is a uniform lattice.

5 Conclusion

We have outlined a unified framework for the whole set of domain decomposition and multilevel methods. The framework has been realized in Diffpack using object-oriented programming techniques. We have indicated that the implementation has the same level of efficiency as tailored, traditional implementations in Fortran or C, but with much more flexibility and extensibility.

REFERENCES

[ABC$^+$97] Arge E., Bruaset A. M., Calvin P. B., Kanney J. F., Langtangen H. P., and Miller C. T. (1997) On the efficiency of C++ for scientific computing. In Dæhlen M. and Tveito A. (eds) *Mathematical Models and Software Tools in Industrial Mathematics.* Birkhäuser.

[ABL97] Arge E., Bruaset A. M., and Langtangen H. P. (1997) Object-oriented numerics. In Dæhlen M. and Tveito A. (eds) *Mathematical Models and Software Tools in Industrial Mathematics.* Birkhäuser.

[Ban94] Bank R. E. (1994) *PLTMG: A Software Package for Solving Elliptic Partial Differential Equations – Users' Guide 7.0.* SIAM Books, Philadelphia.

[BGLV89] Bourgat J. F., Glowinski R., LeTallec P., and Vidrascu M. (1989) Variational formulation and algorithm for trace operator in domain decomposition calculations. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Proc. Second Int. Conf. on Domain Decomposition Meths.*, pages 3–16. SIAM, Philadelphia.

[BL96] Bruaset A. M. and Langtangen H. P. (1996) Object-oriented design of preconditioned iterative methods. *To appear in ACM Trans. Math. Software* .

[BL97] Bruaset A. M. and Langtangen H. P. (1997) A comprehensive set of tools for solving partial differential equations; Diffpack. In Dæhlen M. and Tveito A. (eds) *Numerical Methods and Software Tools in Industrial Mathematics.* Birkhäuser.

[BPS86] Bramble J. H., Pasciak J. E., and Schatz A. H. (1986) The construction of preconditioners for elliptic problems by substructuring, I. *Math. Comp.* 47: 103–134.

[Dif] Diffpack world wide web home page. http://www.oslo.sintef.no/diffpack/.

[Dou95a] Douglas C. C. (1995) Example multigrid code using MPI. ftp://na.cs.yale.edu/pub/mgnet/www/mgnet/Codes/douglas/.

[Dou95b] Douglas C. C. (1995) Madpack: A family of abstract multigrid or multilevel solvers. *Comput. Appl. Math.* 14: 3–20.

[DW90] Dryja M. and Widlund O. B. (1990) Towards a unified theory of domain decomposition algorithms for elliptic problems. In Chan T. F., Glowinski R., Périaux J., and Widlund O. B. (eds) *Proc. Third Int. Conf. on Domain Decomposition Meths.*, pages 3–21. SIAM, Philadelphia.

[DW95] Dryja M. and Widlund O. B. (1995) Schwarz methods of Neumann-Neumann type for three-dimensional elliptic finite element problems. *Comm. Pure Appl. Math.* 48: 121–155.

[GS94] Gropp W. and Smith B. (1994) Scalable, extensible, and portable numerical libraries. In *Proceedings of Scalable Parallel Libraries Conference.* IEEE, Los Alamitos, CA.

[Hac85] Hackbusch W. (1985) *Multi-Grid Methods and Applications.* Springer, Berlin.

[HLM91] Haase G., Langer U., and Meyer A. (1991) Domain decomposition methods with inexact subdomain solvers. *J. Numer. Lin. Alg. Appl.* 1: 27–41.

[Man93] Mandel J. (1993) Balancing domain decomposition. *Comm. Numer. Meth. Engrg.* 9: 233–241.

[PET] Petsc world wide web home page. http://www.mcs.anl.gov/petsc/petsc.html.

[SBG96] Smith B., Bjørstad P., and Gropp W. (1996) *Domain Decomposition. Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, New York.

[Smi91] Smith B. F. (1991) A domain decomposition algorithm for elliptic problems in three dimensions. *Numer. Math.* 60(2): 210–234.

[Xu92] Xu J. (1992) Iterative methods by space decomposition and subspace correction: A unifying approach. *SIAM Review* 34: 581–613.

[Zum96a] Zumbusch G. W. (1996) Multigrid methods in Diffpack. Technical Report STF42 F96016, SINTEF Applied Mathematics, Oslo.

[Zum96b] Zumbusch G. W. (1996) Overlapping domain decomposition methods in Diffpack/ Schur complement domain decomposition methods in Diffpack. Technical report, SINTEF Applied Mathematics, Oslo.

On Object Oriented Programming Languages as a Tool for a Domain Decomposition Method with Local Adaptive Refinement

Brit Gunn Ersland and Magne S. Espedal

1 Introduction and Model Problem

The main objective for this work is to show how Object Oriented programming languages like C++ can simplify the implementation of a complex model where domain decomposition and local adaptive refinement is used. As an example we present a simulator for two phase fluid flow (oil,water) in a porous media, where the library DIFFPACK [Lan94b, Lan94a, Dho] is extensively used. We start by constructing the base classes for the solvers, and use these as building bricks in a more complex system, where different equations are solved on different meshes with domain decomposition on the finest mesh. The decomposed domain is regarded as an array of solvers which compute the solution to an equation on a single domain with appropriate boundary conditions.

For incompressible immiscible displacement of oil by water in a reservoir the following equations yield

$$\nabla \cdot \mathbf{u} = q_1(\mathbf{x}, t) \tag{1.1}$$

$$\mathbf{u} = -\mathbf{K}(\mathbf{x}) M(S, \mathbf{x}) \cdot \nabla p \tag{1.2}$$

$$\phi \frac{\partial S}{\partial t} + \nabla \cdot (f(S)\mathbf{u}) - \epsilon \nabla \cdot (D(S, x) \nabla S) = q_2(\mathbf{x}, t)\,. \tag{1.3}$$

We will use Neumann type of boundary conditions.

u is the total Darcy velocity, which is the sum of the velocity of the oil and water phase. $\mathbf{K}(\mathbf{x})$ is the permeability which depend on the porous medium, $M(S, \mathbf{x})$ denotes

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

the total mobility of the phases,

$$M(S, \mathbf{x}) = \lambda_w(S) + \lambda_o(S),$$

and p is the total fluid pressure. ϕ is the porosity of the porous media which is considered constant. The fractional flow function $f(S)$ is a nonlinear function of the saturation and is given as

$$f(S) = \frac{\lambda_w(S)}{\lambda_w(S) + \lambda_o(S)} \tag{1.4}$$

where the mobility of oil and water,

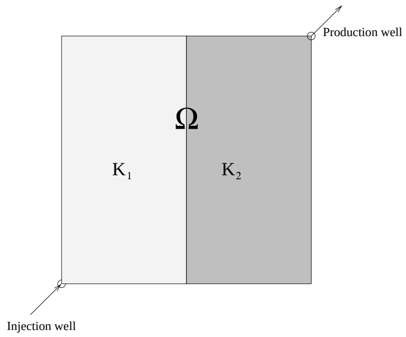
$$\lambda_l, \ l \in \{o, w\}$$

is a given function of S.

The diffusion coefficient $D(S, x)$ depend on the capillary pressure and the permeability, while ε is a small parameter. For a complete survey and justification of the model we refer to [CJ86].

Here, we regard a rectangular domain which consist of two different sediments as depicted in Figure 1. Water is injected in the lower left corner and oil is produced in the upper right corner. Initially we assume that we have an established shock somewhat away from the injection well. A combination of (1.1) and (1.2) give an elliptic equation for the total pressure. This equation is solved by a finite element method on a coarse mesh, then the velocity is derived from the pressure with a second order method [Sæv90].

Figure 1 The figure shows a computational domain Ω with an injection well and a production well.



Since ε is a small parameter the nonlinear saturation equation (1.3) is dominated by the convective part of the equation, which denotes the main transport. It is well known that a nonlinear equation like (1.3) will establish shock like solutions even from smooth initial conditions, therefore we split the function $f(S)$ into two parts [EE87, DR82].

$$f(S) = \bar{f}(S) + b(S)S, \tag{1.5}$$

where $\bar{f}(S)$, the convex hull of the function. multiplied by $\mathbf{u}$ denotes the displacement of an established shock. Hence, we split the saturation equation in two parts. The convective part

$$\Psi(\mathbf{x})\frac{\partial S}{\partial \tau} \equiv \phi\frac{\partial S}{\partial t} + \bar{f}'(S)\mathbf{u}\cdot\nabla S = 0, \tag{1.6}$$

is solved by the Modified Method of Characteristics [DR82]. The elliptic part

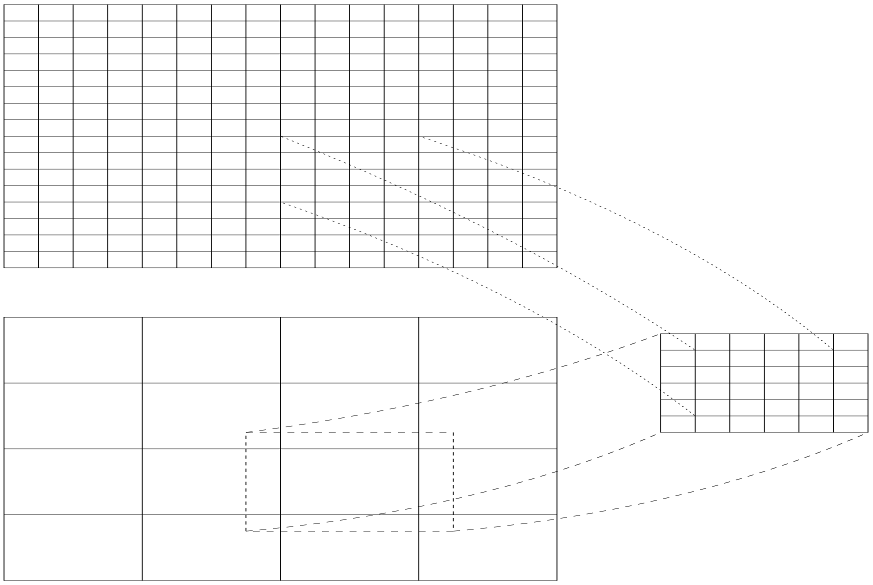
$$\Psi(\mathbf{x})\frac{\partial S}{\partial \tau} + \nabla\cdot(b(S)S\mathbf{u}) - \varepsilon\nabla\cdot(D(S,\mathbf{x})\cdot\nabla S) = 0 \tag{1.7}$$

is solved a finite element method with optimal testfunctions [BM84]. Since the saturation develop shock-like solutions a fine resolution is needed to resolve the shock. However, the elliptic part of the saturation equation (1.7) only contribute to the solution in vicinity of sharp saturation gradients, therefore we want to solve this equation in these areas only.

2 Solution Procedure

Since different resolution is needed for the total pressure and the water saturation, we need to define different grid levels and a mapping between the coarse grid, fine subgrids and a global fine grid. For this problem we have used the mapping shown in Figure 2, where Ω_C is the coarse grid, Ω_F is the global fine grid and the fine subgrids are denoted Ω_m. A sequential timestepping procedure is used [Sæv90], to decouple the equations. Hence, at every time step:

Figure 2 *Mapping between fine grid Ω_F, fine subgrids Ω_m and coarse grid Ω_C .*



```
1. Solve pressure on $\Omega_C$ and determine the velocity on $\Omega_C$.
2. Solve the hyperbolic saturation equation on $\Omega_C$ and on subgrids $\Omega_m$, with
   large saturation gradients.
3. Begin
   for m = 1; number of subdomains do
   if $\Omega_m$ have large gradients
   ∘ update the boundary conditions
   ∘ solve the elliptic equation
   endif
   endo
4. Return to 3 until convergence or a maximum number of iterations is reached.
5. Map solution to $\Omega_F$.
```

Implementation

In this section we will describe the design and some implementation aspects. A main objective is to build a design where each solver may be debugged independently. Now, (1.1), (1.2), (1.6) and (1.7) depend on the mobility function of oil and water, therefor we implemented these functions together with all the methods which uses the mobilities in a separate base class **FracFunc**. The permeability $\mathbf{K}(\mathbf{x})$, which depend on the porous medium is implemented as a separate class. In our test case a domain as depicted in Figure 1 is used, where the permeability depend on (x, y). However, in order to use random permeability or several layers with different sediments only **Permeab** need to be changed.

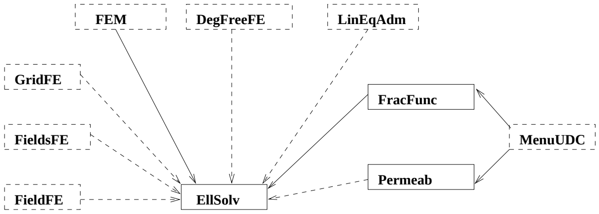
The solution procedure which was described in the previous section requires four solvers, we will just give a brief overview over them before we discuss some details regarding the implementation of the domain decomposition method.

- **Press** - inherit **FracFunc** and contain the data and methods which are needed to solve the pressure equation.
- **Velocity** - inherit **Press**, and use the pressure to derive the velocity.
- **CharSol** - inherit **FracFunc** and use the velocity to integrate backward along the characteristics to solve (1.6). First on the coarce domain Ω_C, then on the fine subdomains Ω_m where the saturation gradient is large. Here the saturation on the uniform fine mesh Ω_F at previous time level is used when we search the new solution on different grid levels.
- **EllSolv** - inherit **FracFunc**, and has a data Type **Permeab**. The class contain all the data and methods which are needed to solve the elliptic part of the saturation equation (1.7) on a single domain. Here, Dirichlet boundary conditions, Neumann type of boundary conditions, or different boundary conditions on different boundaries can be chosen.

Both **Press** and **EllSolv** is derived form the DIFFPACK base class **FEM** [Lan94a] which is a base class for Finite Element programming with DIFFPACK. In addition both **Press** and **EllSolv** has a **LinEqAdm** which is an Abstract Data Type which administers the solvers for a linear system of equations, see figure 3. **FieldFE**, **FieldsFE** and **GridFE** is scalar field, vector field and a finite element grid in the

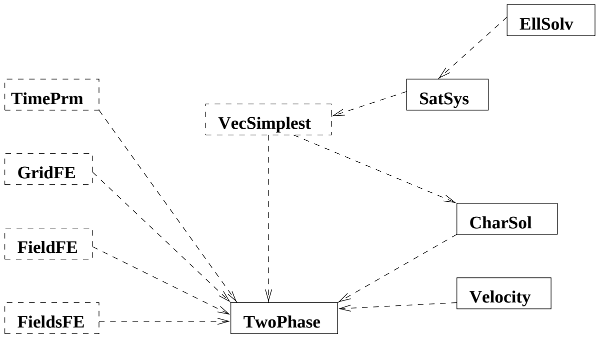
DIFFPACK library. **EllSolv** contains a method which solves the elliptic part of the saturation equation on a given domain Ω_m. Instead of decomposing the computational domain inside **EllSolv** and by this altering the methods in **EllSolv**, we construct a domain decomposition method by making multiple copies of **EllSolv** where the data differs. Now, to use a domain decomposition method like additive or multiplicative Schwarz, some information about the surrounding subdomains are needed. Since the elliptic saturation equation only need to be solved on subdomains where the saturation gradients are large, a boolean variable is needed to indicate if the equation need to be solved on current subdomain at this time level. Therefore, a new class **SatSys** is implemented which has a data type **EllSolv**, an array which keep track of the neighboring subdomains and a boolean variable. In addition SatSys contain methods to initiate the local domain Ω_m and mark boundaries which are at the outer boundary, where Neumann type of boundary conditions are used.

Figure 3 *Data Types and dependencies. Dotted lines indicate a "has a" relationship, while solid lines indicate a "is a" relationship. Solid boxes are newly constructed Data Types, while dotted boxes is included in DIFFPACK.*



Since we want to solve the elliptic saturation by a Schwarz method, we need methods which control our subdomains. To do this we use a general vector in DIFFPACK, **VecSimplest** which can take user defined data types as argument. In our case we use **SatSys** as data type in **VecSimplest**. Let us call the method which solves

Figure 4 *Relationships and dependencies for the two phase reservoir simulator with adaptive local refinement. Dotted lines indicate a "has a" relationship, while the solid lines indicate a "is a" relationship. Solid boxes are newly constructed Data Types, while dotted boxes indicate Data Types from the DIFFPACK package.*



the elliptic saturation equation on a single domain *solveAtThisTimeLevel()* and the vector of subdomain solvers *localSys* while **EllSolv** inside **SatSys** is called *ellsolv.* In our code the method which handles the interior boundary conditions is called *updateLocBoundaries(i)* which mean that the boundary condition for domain i is updated. Hence, our multiplicative Schwarz procedure among refined grids are:

```
for( i=1; i <= nel; i++ ){
if( localSys(i).Active == TRUE) {
updateLocBoundaries(i);
localSys(i).ellsolv.solveAtThisTimeLevel();
   }
}
```

The relationship between the different classes that have been implemented is shown in Figure 4. In order to avoid multiple data, both **EllSolv** and **CharSol** point at the same objects on the same subdomain Ω_m . Likewise do **CharSol** and **Velocity** on the coarse domain Ω_C .

3 Numerical Result

The algorithms which is described in the preveous sections have been tested in 2d on the domain depicted in Figure 1, where the interface between two sediments are inside some of the subdomains Ω_m . The methods which are used to handle the interface conditions are teated in [Ers96] which also contain more numerical results and description of the experiment that is shown here. The results obtained with adaptive local grid refinement in Figure 5 shows good agreement with the results obtained on a single domain, see [Ers96].

We have shown that a domain decomposition method is easy to construct when a object oriented language as C++ is used. Some of the methods which are in **EllSolv**, **Press** and **Velocity** are initially written by K. G. Frøysa, but modified to some extent.

Acknowledgement

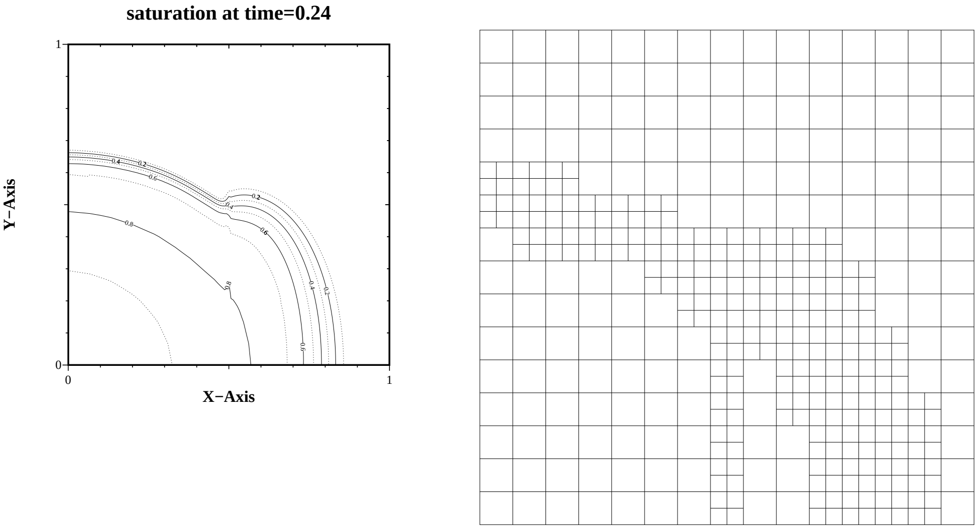
This research was supported by the Norwegian Research Counsel and the University of Bergen.

REFERENCES

[BM84] Barrett J. and Morton K. (1984) Approximate symmetrization and petrov-galerkin methods for diffusion-convection problems. *Computer Methods in Applied Mechanics and Engineering* 45: 97–122.

[CJ86] Chavent G. and Jaffre J. (1986) *Mathematical models and finite elements for reservoir simulation.* North-Holland.

Figure 5 *Computed results with adaptive local grid refinements at time level* $t = 0.24$ *. The refined area are marked with a cross.*



[Dho] http://www.oslo.sintef.no/avd/33/3340/diffpack. The Diffpack WWW home page.

[DR82] Douglas J. and Russell T. (1982) Numerical methods for convection-dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures. *SIAM Journal on Numerical Analysis* 19: 871–885.

[EE87] Espedal M. and Ewing R. (1987) Characteristic petrov-galerkin subdomain methods for two-phase immiscible flow. *Computer Methods in Applied Mechanics and Engineering.* 64: 113–135.

[Ers96] Ersland B. (1996) *On Numerical Methods for Including the Effect of Capillary Pressure Forces on Two-phase, Immiscible Flow in a Layered Porous Medium.* PhD thesis, University of Bergen. Department of Mathematics, University of Bergen.

[Lan94a] Langtangen H. P. (1994) Details of finite element programming in diffpack. Technical report, SINTEF, informatics Oslo.

[Lan94b] Langtangen H. P. (1994) Diffpack: Software for partial differential equations. In Vermeulen (ed) *OON-SKI'94*. Proceedings of the Second Annual Object-Oriented Numerics Conference.

[Sæv90] Sævareid O. (1990) *On Local Grid Refinement Techniques for Reservoir Flow Problems.* PhD thesis, Department of Applied Mathematics, University of Bergen.

Part IV

Applications

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

79

Spectral Element Simulations of Laminar Diffusion Flames

Ullrich Becker-Lemgau and Catherine Mavriplis

1 Introduction

Soot generation is, in the sense of efficiency, a major problem during combustion of hydrocarbons. In order to optimize the design of combustion chambers and avoid soot generation, it is essential to understand the mechanisms of soot growth. In particular, the interactions between chemistry and fluid flow are critically important: experiments have shown that soot production is strongly dependent on the flow situation. The amount of soot generated by laminar diffusion flames is greatly enhanced if the flame becomes unstable [SHJP93, SHS94].

In order to further investigate vortex-chemistry interaction, we develop here a numerical simulation since computations allow us to investigate a wide range of destabilizing effects on laminar flames, both individually and together. In this paper we present spectral element simulations of laminar axisymmetric non-premixed methane-air diffusion flames. The spectral element method [Pat84] is a high-order domain decomposition method which is used in the solution of time-dependent nonlinear partial differential equations. The method has been used in the solution of the Navier-Stokes equations for direct simulation of fluid flow, e.g. [Kar90, FR94]. Here, we extend the Navier-Stokes solver to include chemistry and energy conservation equations. This paper is the first attempt at establishing our simulation capability for laminar diffusion flames. Results are compared with theoretical and experimental flames.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 Governing Equations

The flow underlying an unsteady laminar diffusion flame is a one phase flow with variable density and viscosity due to the temperature change in the flow. It is described by the Navier-Stokes equations:

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot \rho \boldsymbol{v} = 0$$

$$\rho \frac{\partial \boldsymbol{v}}{\partial t} + \rho(\boldsymbol{v} \cdot \boldsymbol{\nabla})\boldsymbol{v} = (\boldsymbol{\nabla} \cdot \mu \boldsymbol{\nabla})\boldsymbol{v} - \boldsymbol{\nabla} p + \boldsymbol{S}_v$$

with velocity $\boldsymbol{v}$, pressure p, density ρ, viscosity μ and a source term $\boldsymbol{S}_v$ which contains the buoyancy force.

The mixing process of fuel and oxygen is described by the mixture fraction ξ which is a conserved scalar for the flow [Spa79]:

$$\rho \frac{\partial \xi}{\partial t} + \rho(\boldsymbol{v} \cdot \boldsymbol{\nabla})\xi = (\boldsymbol{\nabla} \cdot \rho D \boldsymbol{\nabla})\xi$$

with D as the diffusivity. We assume a complete and fast one step reaction from fuel mixed with oxygen to a product mixture without intermediate products and no back reaction.

The energy equation may be expressed in terms of temperature assuming constant heat capacity:

$$c_p \rho \frac{\partial T}{\partial t} + c_p \rho(\boldsymbol{v} \cdot \boldsymbol{\nabla})T = (\boldsymbol{\nabla} \cdot k \boldsymbol{\nabla})T + Q(\xi)$$

with k as conductivity and Q as heat release depending on mixture fraction and flame height. We use experimental data of a methane-air flame for the heat release description [SMP96].

The system of equations is closed by the equation of state:

$$\rho = \frac{M \cdot p}{R \cdot T}$$

where M is the molar weight of the gas mixture and R the gas constant.

3 Spectral Element Model

In order to reduce the complexity of the system of equations we assume at first a flow with constant density and constant properties. So we solve the incompressible Navier-Stokes equations and will let density vary for the buoyancy term only.

The Navier-Stokes equations pose the largest problem since they have nonlinear terms and we need to incorporate the continuity equation. Therefore a multi-fractional time-stepping scheme is introduced which breaks up the equation, treating the nonlinear convection terms separately from the elliptic diffusion terms [OK80]:

$$\frac{\hat{u} - \boldsymbol{u}^n}{\Delta t} = \boldsymbol{u} \cdot \nabla \boldsymbol{u}$$

$$\nabla^2 p = \frac{\rho}{\Delta t} \nabla \cdot \hat{u}$$

$$\frac{\hat{\hat{u}} - \hat{u}}{\Delta t} = -\frac{1}{\rho} \nabla p$$

$$\frac{\boldsymbol{u}^{n+1} - \hat{\hat{u}}}{\Delta t} = \nu \nabla^2 \boldsymbol{u}$$

where $\hat{u}$ and $\hat{\hat{u}}$ are intermediate time step values of the velocities.

First, the nonlinear terms are treated explicitly by a 3rd-order Adams-Bashforth method. Then, the continuity and pressure steps are combined and a Poisson equation for pressure is solved implicitly. Finally, the elliptic velocity diffusion terms are solved implicitly as a Helmholtz problem.

The mixture fraction and temperature equations do not pose such difficulties: the advection terms are easily handled explicitly and we solve also Helmholtz problems for ξ and T. So we have five (in 2D) Helmholtz equations to solve implicitly for velocities (two), pressure, mixture fraction and temperature.

We illustrate the spectral element discretisation for simplicity with the Poisson equation:

$$\nabla^2 u = f.$$

Starting from the Poisson equation we use the weak form (or variational form), i.e. we have to find a solution u which satisfies

$$(\nabla u, \nabla v) = (f, v) \quad \forall v.$$

The physical domain is then subdivided into large elements k upon each of which unknowns u and knowns like f are described as tensor products of high-order orthogonal functions:

$$u_h^k = u_{ij}^k h_i(r) h_j(s),$$

$$f_h^k = f_{ij}^k h_i(r) h_j(s),$$

where h_i's are high-order Lagrangian interpolants based on Legendre polynomials (typically order 7).

So the discrete problem is to find the discrete solution u_h^k on each element satisfying

$$(\nabla u_h^k, \nabla v_h^k)_{GL} = (f_h^k, v_h^k)_{GL} \; \forall v_h^k,$$

where the inner products are done by Gauss-Lobatto quadrature. Finally the contributions from all connecting elements are summed up. This leads to a global matrix equation $Au = Bf$ which is solved by preconditioned conjugate gradient iteration.

4 Validation

To validate our model we first try to compare results to a theoretical solution of a given flame. Several simplifying assumptions are necessary to solve the equations theoretically which will lead to the classical Burke-Schumann flame [BS28].

Figure 1 Burke-Schumann flame: design and flame shapes for over- and underventilated flow situations

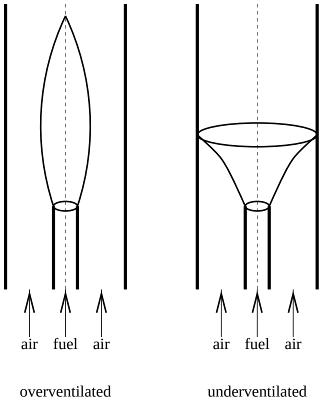
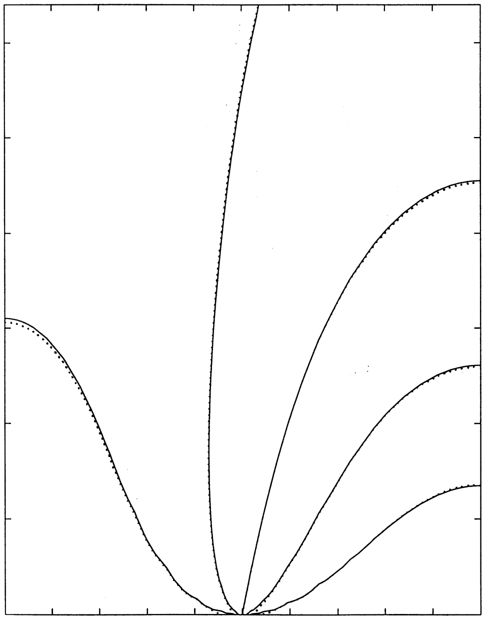


Figure 2 Burke-Schumann flame: comparison of theoretically (dotted lines) and numerically (solid lines) obtained flame shapes for underventilated (leftmost) and overventilated flow situations



Fuel flows through a pipe into a concentric pipe with coflowing air (see Fig. 1). Velocities and properties are assumed to be uniform throughout the domain. The mixing process starts right at the outlet of the fuel pipe and thus the fast one step reaction takes place in a very thin layer.

Two classes of flames are distinguished depending on the fuel type: an overventilated flame where oxygen is in excess and all fuel is consumed by the reaction and an underventilated flame where fuel is in excess and all oxygen is consumed by the reaction. Fig. 1 shows typical flame shapes for both situations.

A comparison of theoretically and numerically obtained flame shapes can be seen in Fig. 2. Due to symmetry only the left half of the pipe is calculated and shown. Fuel is flowing into the domain from the right half of the lower edge. The co-flowing air is entering through the left half. Four overventilated and one underventilated (the leftmost) flame shapes are shown and are found to be in very good agreement with theoretical results.

5 Results

The geometry underlying our calculations is adapted from experiments concerning soot generation in flickering flames [SHJP93, SHS94, SMP96]. It is similar to the geometry of the Burke-Schumann flame (Fig. 1). An axisymmetric tube of diameter 1.1 cm provides the methane fuel flow into a concentric tube of diameter 10.2 cm providing co-flowing air. At the inlet of the fuel pipe a parabolic velocity profile is assumed with a maximum velocity of 12 cm/s. The co-flowing air stream has a uniform velocity of 10.4 cm/s. Because of the axisymmetric design we only have to calculate a two dimensional slice of the pipe.

In the following we will present example calculations representing our approach building a laminar diffusion flame modeling capability.

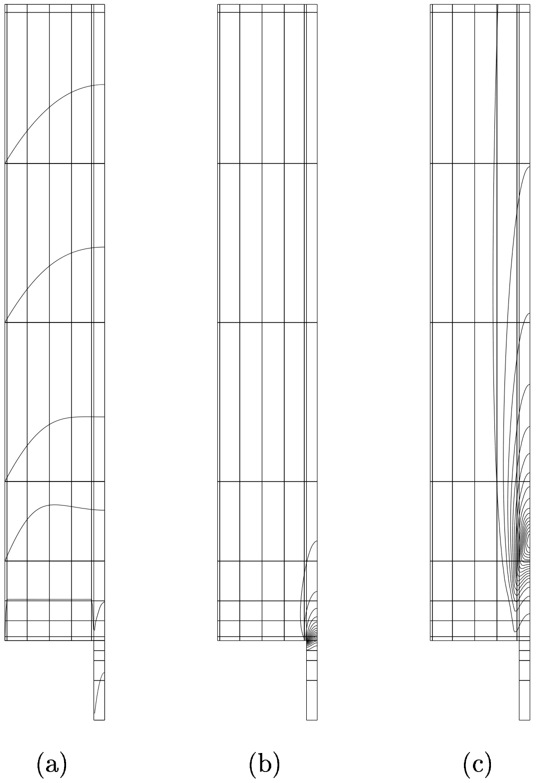
Calculation with Constant Properties

The heavy coupling of the equations, especially the heat release term in the energy equation and the buoyancy term in the vertical momentum equation, makes it difficult to arrive at a suitable base solution from which we may start pulsing the flame. Therefore we start our calculations with a steady, incompressible flame and constant properties: $\rho = 0.1\,\mathrm{kg/m^3}$, $\mu = 2 \cdot 10^{-5}\,\mathrm{kg/m\,s}$, $c_p = 1000\,\mathrm{J/kg\,K}$, $k = 0.01\,\mathrm{J/m\,s\,K}$, $D = 1 \cdot 10^{-4}\,\mathrm{m^2/s}$. These values are chosen to fulfill the requirement of equal Prandtl and Schmidt numbers for diffusion flames. The Reynolds number based on the average velocity of 10 cm/s and the diameter of the air stream is $\mathrm{Re} = 51$. Because of the constant density no buoyancy is taken into account and the calculated temperatures have no influence on the fluid flow. For the heat release function in the temperature equation a simplified linear model is chosen.

The calculation domain is split into 67 spectral elements (see Fig. 3). The geometrical discontinuities between the fuel and air tubes require a domain decomposition with fine resolution around this interface region. So two narrow slices of elements are placed along the air inlet and between the air and fuel streams.

Fig. 3 (a) shows calculated axial velocity profiles at several cross-sections. The two

Figure 3 Results for calculation with constant properties: (a) profiles of axial velocity, (b) contour lines for mixture fraction ξ, (c) contour lines for temperature distribution



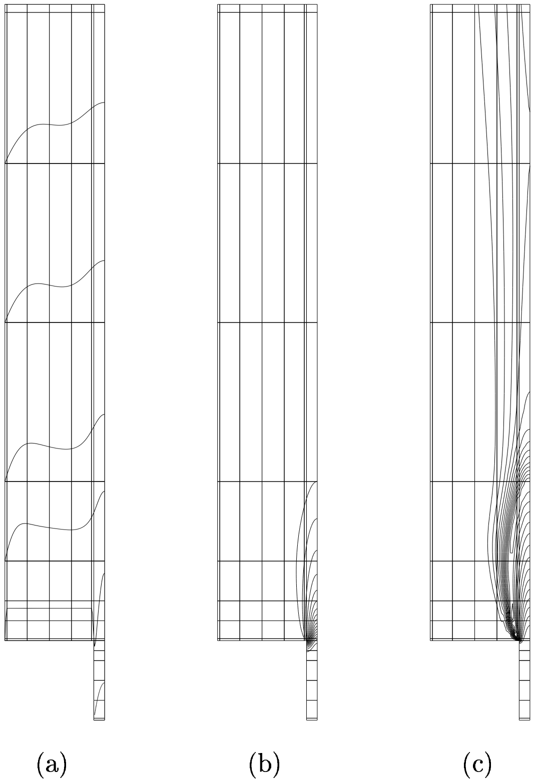
different profiles of the incoming fuel and air streams level out very quickly and a regular pipe flow profile is obtained towards the outlet.

The contour lines for the mixture fraction distribution (Fig. 3 (b)) imply a fast diffusion of oxygen into the fuel stream. The outmost contour line represents the value of ξ for which methane and air are in a stoichiometric proportion. Because we assume a fast and one step reaction, this line indicates the flame shape where both fuel and oxygen are completly consumed by the reaction. The flame height is 5 cm, that is somewhat smaller than the observed flame height of 8 cm from the experiments [SMP96] due to our simplifying assumptions.

The temperature distribution (Fig. 3 (c)) is typical for a laminar diffusion flame. High temperatures are reached along the flame shape and especially at the flame tip.

As a first approach to calculate an unsteady flame we took this calculation and pulsed the fuel flow with a frequency of 10 Hz. But the results did not show any recirculation or separation as had been observed during the experiments. The

Figure 4 Results for calculation with variable properties: (a) profiles of axial velocity, (b) contour lines for mixture fraction ξ, (c) contour lines for temperature distribution



calculated flame height was increased to 7 cm.

Calculation with Variable Properties and Buoyancy

In this case we still assume a steady and incompressible flow with $\rho = 0.25\,\mathrm{kg/m^3}$ and $c_p = 1000\,\mathrm{J/kg\,K}$ but introduce temperature dependent properties, the models for which can be found in [SG91]:

$$\mu = 1.85 \cdot 10^{-5} \left(\frac{T}{T_0}\right)^{0.7} \frac{\mathrm{kg}}{\mathrm{m\,s}}, \qquad k = \frac{\mu c_p}{\mathrm{Pr}}\ \frac{\mathrm{J}}{\mathrm{m\,s\,K}}, \qquad D = \frac{\mu}{\rho\ \mathrm{Sc}}\ \frac{\mathrm{m}^2}{\mathrm{s}}$$

where the Prandtl number Pr and Schmidt number Sc are both equal 0.75 and T_0 denotes the room temperature. This gives a Reynolds number of 143 for $T = T_0$ based on velocity and diameter of the air stream. In this calculation we do not solve a temperature equation but, instead, use a temperature function $T(\xi)$ derived from experiments [SMP96] to ensure realistic temperature values.

Variable density is introduced only for the buoyancy term. The temperature in the equation of state (see section 2) is kept constant. So the density in the buoyancy term varies only with mixture fraction in the fluid flow and pressure.

As expected, we get a speed up due to buoyancy which can be recognized by axial velocity profiles (Fig. 4 (a)). The maximum speed in the flame region is 25 cm/s. The flame height, indicated by the outmost contour line of Fig. 4 (b), is 8 cm and exactly the same as the observed flame height in the experiments.

The temperature distribution (Fig. 4 (c)) looks very similar to the one from the first calculation. But the region of the highest temperature is now spread around the flame shape and not only at the flame tip. This is caused by the temperature mixture fraction relation we used in this calculation.

6 Conclusion

We have shown that the spectral element method is applicable for investigations of laminar diffusion flames. Thus we have widened the range of applications for this method.

Our model was validated by very good agreement between theoretical and numerical results for the Burke-Schumann flame. For the steady flame the model is already suitable to predict flame height very well if buoyancy and variable properties are taken into account. However, calculations for the unsteady flame have not led to comparable results so far.

Further improvements are necessary for realistic predictions of the unsteady case. We have to introduce buoyancy and temperature dependent properties as for the steady flame. Also reexamination of the heat release treatment and other assumptions of the model are needed for realistic temperature distributions.

These extensions to our model will hopefully improve our results for the unsteady flickering diffusion flame in the near future.

Acknowledgement

This work was partially supported by the George Washington University Research Enhancement Fund. Partial travel support was provided by the US National Science Foundation under Grant ECS-9209347. Support from both sources is gratefully acknowledged.

The authors would like to thank Prof. R. Goulard, Prof. J.H. Miller and Dr. M. Pivovarov from The George Washington University for fruitful discussions.

REFERENCES

[BS28] Burke S. and Schumann T. (1928) Diffusion flames. *Indust. Eng. Chem.* 20: 998–1004.

[FR94] Fischer P. and Rønquist E. (1994) Spectral element methods for large scale parallel navier-stokes calculations. *Computer Methods in Applied Mechanics and Engineering* 116: 69–76.

[Kar90] Karniadakis G. (1989/90) Spectral element simulations of laminar and turbulent flows in complex geometries. *Applied Numerical Mathematics* 6: 85–105.

[OK80] Orszag S. and Kells L. (1980) Transition to turbulence in plane poiseuille and plane couette flow. *Journal of Fluid Mechanics* 96: 159–205.

[Pat84] Patera A. (1984) A spectral element method for fluid dynamics: Laminar flow in a channel expansion. *Journal of Computational Physics* 54(3): 468–488.

[SG91] Smooke M. and Giovangigli V. (1991) Formulation of the premixed and nonpremixed test problems. In Smooke M. (ed) *Reduced Kintetic Mechanisms and Asymptotic Approximations for Methane-Air Flame*, volume 384 of *Lecture Notes in Physics*, pages 1–28. Springer-Verlag, Berlin.

[SHJP93] Smyth K., Harrington J., Johnsson E., and Pitts W. (1993) Greatly enhanced soot scattering in flickering CH_4/air diffusion flames. *Combustion and Flame* 95: 229–239.

[SHS94] Shaddix C., Harrington J., and Smyth K. (1994) Quantitative measurements of enhanced soot production in a flickering methane/air diffusion flame. *Combustion and Flame* 99: 723–732.

[SMP96] Smooke M., Miller J., and Pivovarov M. (1996) Private communications.

[Spa79] Spalding D. (1979) *Combustion and Mass Transfer.* Pergamon Press, Oxford.

80

Variable-degree Schwarz Methods for Unsteady Compressible Flows

Xiao-Chuan Cai, Charbel Farhat, and Marcus Sarkis

1 Introduction

We introduce a new variant of the overlapping Schwarz method (OSM) for solving unsteady problems. In particular we study implicit methods ([FFL93, FS89, VM95]) for obtaining the time accurate solution of the compressible Navier-Stokes equations discretized on two-dimensional unstructured meshes. When using implicit methods, a large, sparse linear system must be constructed and solved at each time step. Depending of the size of the time step, and several other flow parameters, the conditioning of the matrix may change from well-conditioned to mildly ill-conditioned. Furthermore, due to the complexity of the flow pattern, at a given time step the matrix may be ill-conditioned in certain subregions, for example near the airfoil, and relatively well-conditioned elsewhere. To solve these systems iteratively, it is necessary to have a family of preconditioners, such as OSM, whose strength can be adjusted locally in each subdomain according to the flow condition.

It is known that when constructing a preconditioner for a single linear system $Au = f$ all the information needed is from the matrix A. However, the issue for time dependent problems is different. A sequence of interrelated systems $A^{(k)}u^{(k)} = f^{(k)}$ have to be solved. If the matrix, especially in its (often inexactly) factorized form, obtained at a previous time step can be properly used, then the preconditioner at the current time step can be obtained cheaply. More precisely, at each time step, we solve the linear system by a preconditioned GMRES method and in the preconditioning stage, following the general OSM framework, we solve the local subdomain problems by another preconditioned GMRES method with different preconditioners and stopping conditions. In each subdomain the preconditioner is built by using a polynomial in two matrix variables, namely the matrix, in its *unfactorized* form, of the current time step k and another matrix, in its *factorized* form obtained at a previous time step j. The degree of the matrix polynomial reflects the conditioning of the subdomain matrix. Note that classical Schwarz methods correspond to the case where the degree of the matrix polynomials always equals to one. In our new method, the degree of the polynomial varies from subdomain to subdomain depending on the flow conditions,

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

and therefore we refer to the methods as variable degree Schwarz methods (VDS).

In this paper, we also study the effects of the overlapping size, the number of subdomains, and the inexact subdomain solvers. Since the construction of the preconditioner is expensive, we also explore the possibility of reusing the preconditioner for several time steps.

2 Variable-degree Schwarz Methods

Suppose that at each time step k we need to solve $A^{(k)}u^{(k)} = f^{(k)}$ by an iterative method with a preconditioner $M^{(k)}$ to a certain accuracy, i.e.,

$$\left\| M^{(k)} \left(A^{(k)}u^{(k)} - f^{(k)} \right) \right\|_2 \leq \tau \| M^{(k)} f^{(k)} \|_2, \tag{2.1}$$

where τ is a given tolerance. Let n be the total number of unknowns and $\mathcal{N} = \{1, \cdots, n\}$. To define algebraic Schwarz algorithms, see e.g. [CS96], we first partition $\mathcal{N}$ into n_0 nonoverlapping subsets $\{\mathcal{N}_i\}$ whose union is $\mathcal{N}$. To generate an overlapping partitioning with overlap *ovlp*, we expand each subgrid $\mathcal{N}_i$ by *ovlp* number of neighboring nodes, denoted as $\tilde{\mathcal{N}_i}$. We denote by L_i the vector space spanned by the set $\tilde{\mathcal{N}_i}$. For each subspace L_i, we define an orthogonal projection operator I_i and $A_i^{(k)} = I_i A^{(k)} I_i$, which is an extension to the whole subspace, of the restriction of $A^{(k)}$ to L_i. We define its "inverse" by $(A_i^{(k)})^{-1} \equiv I_i \left((A_i^{(k)})_{|L_i} \right)^{-1} I_i$. The classical additive and multiplicative Schwarz algorithms can be described as follows([CS96, CM94, DW94]): Solve $MA^{(k)}u^{(k)} = Mf^{(k)}$ by a Krylov subspace method, where

$$M = (A_i^{(k)})^{-1} + \cdots + (A_{n_0}^{(k)})^{-1} \ , \quad \text{and} \tag{2.2}$$

$$M\,A^{(k)} = I - \left(I - (A_i^{(k)})^{-1} A^{(k)} \right) \cdots \left(I - (A_{n_0}^{(k)})^{-1} A^{(k)} \right) \tag{2.3}$$

for the additive and multiplicative Schwarz algorithms, respectively.

There are three major steps in the construction of the Schwarz preconditioners, namely 1) the construction of the matrix $A^{(k)}$; 2) the construction of the matrices $A_i^{(k)}$; and 3) the incomplete factorization of the matrices $A_i^{(k)}$. In fact Step 1) is not necessary since the matrices constructed in Step 2) can be used to calculate the matrix-vector multiplications. Since we are interested in implicit methods, Step 2) has to be done at every time step no matter how expensive it is. One expensive step in the construction of the preconditions as formulated above for time dependent problems is Step 3). One way to avoid the frequent factorization of $A_i^{(k)}$ is to simply use some old factorized matrix $A_i^{(j)}$ calculated at time step j, where $j < k$. However, this method may not be very effective if j and k are too far apart. More discussion on using frozen preconditioners can be found later in the paper.

Another problem with the Schwarz preconditioners (2.2) and (2.3) is that all subdomains are treated equally in terms of the level of preconditioning in the sense that the number of applications of $(A_i^{(k)})^{-1}$, or its inexact version, is the same on all subdomains, regardless of the fact that the subdomain matrices $A_i^{(k)}$ have vary different condition numbers. Physically speaking, the behavior of the flows in subdomains near the body of the airfoil, or near the shocks, is very different from the other regions. More preconditioning is needed only in subdomains where the real action take place.

We propose a method that places different levels of preconditioning in different subdomains and will also show by numerical experiments that the methods remain to be effective even if j and k are far apart from each other. The idea is simple. We replace the matrix-vector multiply in (2.2) or (2.3)

$$w = (A_i^{(k)})^{-1} v \tag{2.4}$$

by another *iterative procedure* with $(B_i^{(j)})^{-1}$ as the preconditioner. Here $B_i^{(j)}$ is an incomplete factorization of $A_i^{(j)}$ with certain levels of fill-in at time step j. More precisely speaking, to obtain w for a given v, we run several steps of GMRES in the subspace L_i such that

$$\left\| (B_i^{(j)})^{-1} (v - A_i^{(k)} \tilde{w}) \right\|_2 \leq \delta \left\| (B_i^{(j)})^{-1} v \right\|_2 . \tag{2.5}$$

We then set $w := \tilde{w}$. Here δ is a pre-selected small value. Examples can be found in Σ3. In the matrix language, we replace the matrix $(A_i^{(k)})^{-1}$ in (2.4) by a matrix polynomial $poly_i \left((B_i^{(j)})^{-1} A_i^{(k)} \right)$ of a certain degree. The actual degree depends on the number of GMRES iterations needed in the subspace L_i. To put them into a single form, the additive Schwarz preconditioner becomes

$$M = poly_1 \left((B_1^{(j)})^{-1} A_1^{(k)} \right) + \cdots + poly_{n_0} \left((B_{n_0}^{(j)})^{-1} A_{n_0}^{(k)} \right) .$$

Note that this preconditioner does not contain $(A_i^{(k)})^{-1}$, but it contains certain spectral information from $(A_i^{(k)})^{-1}$. This makes it very effective. In fact, M is a truncated series representation of $(A_i^{(k)})^{-1}$ based on a splitting of $A_i^{(k)}$ into the sum of $B_i^{(j)}$ and $A_i^{(k)} - B_i^{(j)}$. A discussion on a related polynomial preconditioning method can be found in [GO93]. We note that in a given subdomain, the number of GMRES iterations, or the degree of the polynomial, is determined by the conditioning of the local stiffness matrix. The multiplicative version can be constructed in a similar way.

We remark that since the preconditioner changes in the GMRES loop due to the stopping condition determined by δ, it is generally more appropriate to use the flexible GMRES [Saa93], which is slightly more expensive than the regular one. We do not use the flexible GMRES in this paper since the regular GMRES presents no problem for our test cases.

3 Numerical Results

The goal of this section is to demonstrate the usefulness of the family of VDS preconditioners in the implicit solution of compressible flow problems. We apply our algorithms to the simulation of two-dimensional low Reynolds number flows past a

NACA0012 airfoil at high angle of attack ($30^{\rm deg}$) and two different Mach numbers. No steady state solutions exists for both test cases described below. **Test 1**: The subsonic case with $M_\infty = 0.1$ and $Re = 800.0$. We use a pre-generated shape regular triangular mesh, Mesh12k, with 12280 nodes. **Test 2**: The transonic case with $M_\infty = 0.84$ and $Re = 1600.0$. We use a mesh, Mesh48k, with 48792 nodes obtained by uniformly refining the mesh used in **Test 1**. Because of the page limit, we do not discuss the discretization, time stepping and mesh partitioning in this paper; interested reader should consult [CFS96] for details.

In the implementation of VDS, we partition the mesh by using the recursive spectral bisection method. The sparse matrix is constructed at every time step, and stored in the Compressed Sparse Row format. The subdomain matrices are obtained by taking elements, according to a pre-selected index set, from the global matrix. A symbolic ILU(0) factorization of the subdomain matrix is performed at the very first time step, and reused at all the later time steps. This is possible due to the fact that the matrices, constructed at every time step, share the same non-zero pattern. We also tested the ILU(k) ($k > 0$) preconditioners, which are not competitive with ILU(0) in terms of the CPU time in our implementation for both test cases. We remark that if ILU with drop tolerance is used then the non-zero pattern of the matrices may change and the previously obtained symbolic factorizations cannot be reused.

We note that at the beginning of the motion of the flow, i.e., when the non-dimensionalized time $t \leq 1.0$, the flow changes so drastically that the use of any time step size δt^n that makes the corresponding CFL number larger than 1.0 would result in the loss of time accuracy for the entire calculation. This implies that small δt^n have to be used when $t \leq 1.0$, and therefore, the implicit method has to be abandoned for this initial period of time. In our experiments, the implicit solver is turned on at $t = 1.0$. The solution for the period $0 < t \leq 1.0$ is obtained with an explicit method with CFL=0.8.

The reports given below are based on running our implicit methods for 100 time steps starting at $t = 1.0$. We shall use **MaxIt** to denote the maximum number of global GMRES iterations and **TotalIt** the total number of global GMRES iterations within this 100 linear system solves. To measure the approximate cost of the methods, we use **EMatVec** to denote the equivalent number of global matrix-vector multiplications, which includes the actual stiffness matrix-vector multiplications and the preconditioning-matrix-vector multiplications.

Let us first discuss the dependence of the convergence rate on the number of subdomains. We use 5 different decompositions of Ω, with both Mesh12k and Mesh48k. The number of subdomains goes from 8 to 128. We run both **Test 1** and **2**, with *ovlp* equals to one fine mesh cell. In Table 1, we present the maximum number of global GMRES iterations within one hundred time steps and its corresponding **EMatVec**. If multiplicative VDS is used even without the special subdomain coloring or ordering, **MaxIt** is independent of the number subdomains for reasonably large number of subdomains, such as 128. An interesting case is shown on the top left portion of Table 1 which indicates that if additive VDS is used for the subsonic problem the number of maximum iterations does grow, though not very fast, as the number of subdomains becomes large. In this case, we believe that a coarse space may be useful to reduce the dependence on the number of subdomains. However, we have not implemented the coarse grid solver yet. For transonic problems, our tests show that the use of a coarse level grid is not necessary with both additive and multiplicative VDS preconditioners.

Table 1 CFL=50, $\tau = 10^{-3}$, $ovlp = 1$. We use GMRES/ILU(0) as inexact local solvers with $\delta = 10^{-1}$.

ASM	**Test 1**			**Test 2**		
# subdomains	MaxIt	TotalIt	EMatVec	MaxIt	TotalIt	EMatVec
ASM 8	6	545	3150	6	519	1471
ASM 16	7	585	3529	6	506	1628
ASM 32	9	673	3851	6	560	1864
ASM 64	10	756	4223	6	600	2184
ASM 128	11	842	5621	7	603	2192
MSM 8	4	292	1613	3	300	832
MSM 16	4	316	1812	3	300	915
MSM 32	4	320	1834	3	300	994
MSM 64	4	344	1900	3	300	1089
MSM 128	4	351	2335	3	300	1084

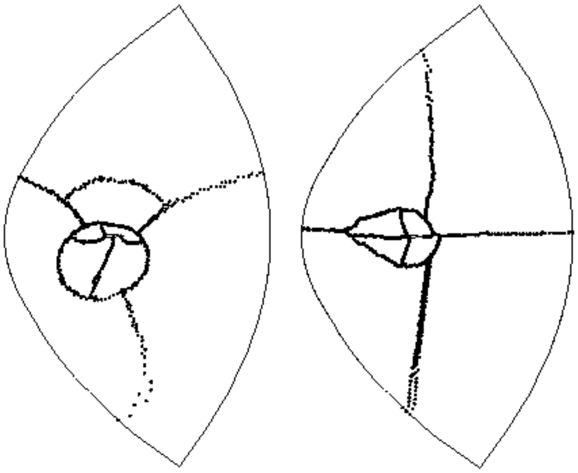
Table 2 Global GMRES/(multiplicative VDS) with CFL=50, $\tau = 10^{-3}$, $\delta = 10^{-1}$, $ovlp = 1$ and the local solvers are GMRES/ILU(0).

	Ω_1	Ω_2	Ω_3	Ω_4	Ω_5	Ω_6	Ω_7	Ω_8
Test 1, MaxIt	2	2	2	3	4	6	3	5
TotalIt	150	106	113	225	307	597	291	421
Test 2, MaxIt	2	2	2	2	2	2	1	1
TotalIt	127	200	109	107	132	185	150	100

Whether overlap is useful or not is a rather subtle issue. It depends on the global linear stopping parameter τ defined in (2.1) and the local linear stopping parameter δ defined in (2.5). According to a large number of tests we did large overlaps can reduce the number of iterations and CPU time only if the stopping parameter δ is small. In our situation when $\tau = 10^{-3}$, we find $\delta = 10^{-1}$ offers the best CPU results, and therefore we do not need large overlaps. In the rest of the tests, we use this set of τ and δ, and $ovlp = 1$.

We next look at the degree of preconditioning polynomial in each subdomain. We focus on the 8 subdomain cases with GMRES/ILU(0) as local subdomain solvers. The partitionings used for Mesh12k and Mesh48k are different, as shown in Fig. 1. The subdomains are numbered as in Fig. 1. The results obtained for one hundred time steps starting at $t = 1.0$ are summarized in Table 2. It turns out the required degrees of local preconditioning polynomials are quite different. For the subsonic case, subdomains Ω_6 and Ω_8 need more iterations (4 and 6 respectively) than other subdomains. The left picture of Fig. 1 shows that these two subdomains cover the top portion of the airfoil.

Figure 1 Left figure shows the partitioning of Mesh12k into 8 subdomains and right one shows that for Mesh48k. The airfoil is at the center of the domain.



Only two iterations are needed for subdomains that are far away from the airfoil, such as Ω_1, Ω_2 and Ω_3. The number of iterations reflects the conditioning of the subdomain matrix. For the transonic case, all subdomains need either one or two iterations. Tables 1 and 2 also show that the number of global and local iterations are surprisingly small. This indicates that the linear systems of equations are in fact not too ill-conditioned. We believe that this is due to the use of relatively small time steps, which is necessary in order to obtain time accurate solutions.

Table 3 Global GMRES/(multiplicative VDS) with CFL=50, $\tau = 10^{-3}$, $\delta = 10^{-1}$, $ovlp = 1$. The number of subdomains is 8. For the FreezeIt=200 case, the numbers are taken for 200 time steps divided by 2.

FreezeIt=	1	5	10	50	100	200
Test 1, EMatVec	1613	1611	1615	1618	1629	1875
TotalIt	292	292	291	290	291	321
Test 2, EMatVec	832	829	830	842	868	1243
TotalIt	300	300	300	305	315	469

Finally, we examine the effect of using the same preconditioner, or part of the preconditioner, for several time steps without doing the factorization at every time step. In Table 3, we summarize the results for using different numbers of frozen steps, namely FreezeIt $= 1, 5, \ldots$. There is a range of optimal FreezeIt one can choose from; similar numbers of EMatVec are obtained in our implementation for FreezeIt ranging from 5 to 50. For the subsonic case, we can go a bit further, e.g., take FreezeIt = 100.

4 Conclusions

We proposed and tested a family of variable degree Schwarz(VDS) preconditioned GMRES methods for solving linear systems that arise from the discretization of unsteady, compressible N.-S. equations on 2D unstructured meshes for both subsonic and transonic flows past a single element NACA0012 airfoil. In VDS, the level of preconditioning in each subdomain varies according to the local flow condition, therefore extra preconditioning is performed only when and where it is needed. For subsonic problems, we found that the conditioning of the subdomain matrices changes quite a bit from one flow region to another, and extra local preconditioning in subdomains in which the flow changes drastically can significantly reduce the total number of global linear iterations. This is somewhat less obvious for transonic flow, which needs a nearly uniformly small global and local number of iterations. When using VDS, the best results are obtained with small overlap. For the multiplicative version, the convergence rate depends very mildly on the number of subdomains (up to 128 subdomains have been tested), and for the additive version, a slight dependence is observed for the subsonic test problem and therefore a coarse space might be useful.

Acknowledgement

The first author was supported in part by NSF ASC-9457534, NSF ASC-9217394 and NASA NAG5-2218. The second author was supported in part by NSF ASC-9217394, NSF ASC-9217394 and NASA NAG5-2218. The third author was supported in part by NSF ASC-9406582, NSF ASC-9217394 and NASA NAG5-2218.

REFERENCES

[CFS96] Cai X.-C., Farhat C., and Sarkis M. (1996) Variable-degree Schwarz methods for the implicit solution of unsteady compressible Navier-Stokes equations on two-dimensional unstructured meshes. Technical Report ICASE Report No. 96-48, ICASE, NASA Langley Research Center.

[CM94] Chan T. F. and Mathew T. P. (1994) Domain decomposition algorithms. *Acta Numerica* (61-143).

[CS96] Cai X.-C. and Saad Y. (1996) Overlapping domain decomposition algorithms for general sparse matrices. *Numer. Lin. Alg. Applics* 3: 221–237.

[DW94] Dryja M. and Widlund O. B. (1994) Domain decomposition algorithms with small overlap. *SIAM J. Sci. Comp.* 15(3): 604–620.

[FFL93] Farhat C., Fezoui L., and Lanteri S. (1993) Two-dimensional viscous flow computation on the Connection Machine: Unstructured meshes, upwind schemes and parallel computation. *Comput. Methods Appl. Mech. Engrg.* 102: 61–88.

[FS89] Fezoui L. and Stoufflet B. (1989) A class of implicit upwind schemes for Euler simulations with unstructured meshes. *J. Comp. Phys.* 84: 174–206.

[GO93] Golub G. and Ortega J. M. (1993) *Scientific Computing: An Introduction with Parallel Computing*. Academic Press, Inc.

[Saa93] Saad Y. (1993) A flexible inner-outer preconditioned GMRES algorithm. *SIAM J. Sci. Stat. Comput.* 14: 461–469.

[VM95] Venkatakrishnan V. and Mavriplis D. J. (Aug. 1995) Implicit method for the computation of unsteady flows on unstructured grids. Technical Report ICASE

Report No. 95-60, ICASE, NASA Langley Research Center.

81

Adaptive Zonal Recognition for Viscous/Inviscid Coupling

A. M. Cuffe, C.-H. Lai, and K. A. Pericleous

1 Introduction

The Navier-Stokes equation is the model generally used to describe the flow of a viscous fluid. In its incompressible form this model is written as

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\underline{u}\phi) = \nu \nabla^2 \phi - \frac{1}{\rho} \nabla p, \tag{1.1}$$

$$\nabla \cdot \underline{u} = 0, \tag{1.2}$$

where $\phi = u, v$ and $\underline{u} = (u, v)^T$.

This system can be computationally demanding. As is well known, high Reynolds number laminar flows produce a boundary layer that is thin compared to the overall flow domain. As a result, time may be wasted by solving the Navier-Stokes equations across the whole domain when the Euler equations would be more appropriate in a large proportion of the domain.

A solution is to divide the flow into viscous and inviscid regions and solve these two regions separately using a suitable iterative technique. This is known as viscous/inviscid interaction [Kno86], and requires zones to be manually predefined [SH86].

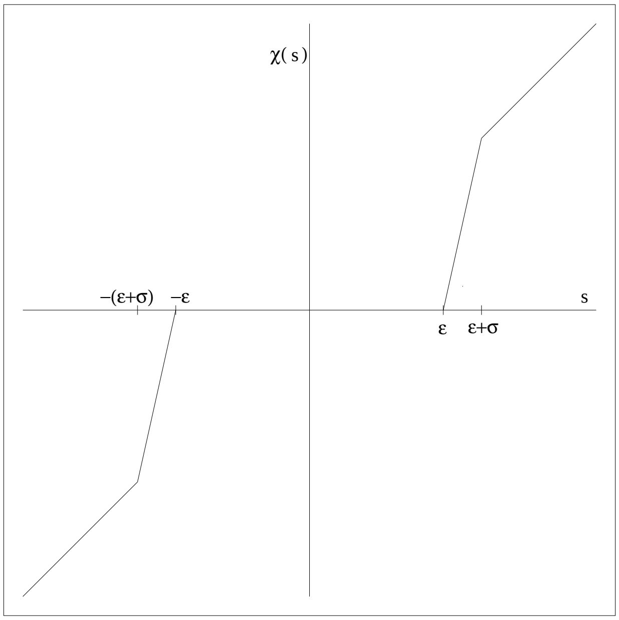
This paper describes the development of an adaptive zonal recognition procedure that does not require manual partitioning and therefore overcomes many of the disadvantages of early zonal methods.

2 Zonal Recognition

The correct choice of zonal boundaries is very important in viscous/inviscid coupling in order to achieve an efficient solution. Consequently there is considerable interest in developing zonal recognition techniques. Work by Perkins and Rodrigue in 1989 [PR89]

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Figure 1 The χ function with a straight line in the transition region



involved computing finite difference value of the viscous term at discrete points. In the same year Brezzi, Canuto and Russo [BCR89] developed a zonal recognition function called the χ-method. More recently Margot [Mar93] developed a physically guided zonal approach. This involved running an initial course grid problem and examining the magnitude of the viscous terms in order to give an indication of the best position for the zonal boundaries. Current work by the authors aims to develop an adaptive domain decomposition technique with zonal recognition and decoupling within a single code framework.

The development of the zonal recognition technique is based on the χ-method by Brezzi, Canuto and Russo. The χ-method can be considered as a truncation technique that reduces the Navier-Stokes system to the Euler in regions where the viscous term is small. This is done by replacing the viscous term $\nabla^2\phi$ in the Navier-Stokes equation by a function $\chi(\nabla^2\phi)$. This function coincides with $\nabla^2\phi$ when the viscous term is large and equates to zero when its value is small, thus becoming the Euler equation.

In particular, the χ function described by Brezzi may be written,

$$\chi(s) = \begin{cases} 0, & |s| \leq \epsilon \\ f(s), & \epsilon < |s| < \epsilon + \sigma \\ s, & |s| \geq \epsilon + \sigma \end{cases} \tag{2.3}$$

where s is, in this case, the viscous term. The values ϵ and σ are threshold parameters which define the size of the viscous and transition regions respectively. By taking a strictly increasing function in the transition region between ϵ and $\epsilon + \sigma$ the χ function becomes a monotonically increasing, continuous function. Brezzi chose a straight line in this region (see Figure 1), while Arina and Canuto used a third-degree polynomial [AC93].

Brezzi originally applied the χ-method in a finite difference context and preliminary results for a one-dimensional test problem have shown that this method works well using a finite difference discretisation. More recently, Achdou and Pironneau [AP93] have applied the χ-method in a finite element method. We have incorporated the χ-method in a finite volume context [LCP96].

3 A Truncation Technique for Finite Volume Methods

Solving the Navier-Stokes equation using a finite volume method involves the integration over a control volume Ω. The Navier-Stokes equation then becomes,

$$\int\!\!\int_{\Omega} \frac{\partial \phi}{\partial t}\, d\Omega + \int\!\!\int_{\Omega} \nabla \cdot (\underline{u}\phi)\, d\Omega = \int\!\!\int_{\Omega} \nu \nabla^2 \phi \, d\Omega - \int\!\!\int_{\Omega} \frac{1}{\rho} \nabla p \, d\Omega, \tag{3.4}$$

which can be rewritten as

$$\int\!\!\int_{\Omega} \frac{\partial \phi}{\partial t}\, d\Omega + \int_{\partial\Omega} (\underline{u}\phi) \cdot \underline{n}\, ds = \int_{\partial\Omega} \nu (\nabla \phi \cdot \underline{n})\, ds - \int\!\!\int_{\Omega} \frac{1}{\rho} \nabla p \, d\Omega. \tag{3.5}$$

It is difficult to truncate the diffusion term in this form as it is now represented by a surface integral. Therefore a modification to the original truncation method is required. The contribution to viscous effect comes from the shear stress at the cell faces, therefore it is reasonable to apply the truncation method to the velocity gradients. Thus, the equation becomes

$$\int\!\!\int_{\Omega} \frac{\partial \phi}{\partial t}\, d\Omega + \int_{\partial\Omega} (\underline{u}\phi) \cdot \underline{n}\, ds = \int_{\partial\Omega} \nu \ \underline{\chi}(\nabla \phi) \cdot \underline{n}\, ds - \int\!\!\int_{\Omega} \frac{1}{\rho} \nabla p \, d\Omega \tag{3.6}$$

where $\underline{n}$ denotes the unit normal vector, and the vector truncation method is denoted by

$$\underline{\chi}(\nabla \phi) \ = \ (\chi(\frac{\partial \phi}{\partial x}) \ , \ \chi(\frac{\partial \phi}{\partial y}))^T. \tag{3.7}$$

The vector truncation method allows a smooth transition of mathematical models from Navier-Stokes to parabolised Navier-Stokes and then to Euler.

This method has been successfully implemented in an in-house two-dimensional finite volume Navier-Stokes code that uses a cell-centred approach. Early studies have shown that the the vector truncation method gives numerical results similar to those obtained by using the finite volume method, the error being dependent on the choice of parameters ϵ and σ. By adjusting these parameters a solution can be obtained which is closer to the finite volume solution. A number of numerical tests have been performed with varying values for the two parameters ϵ and σ on a flat plate problem and an aerofoil problem [LCP96].

From these tests, it has become apparent that keeping the value of ϵ small reduces the error $\|u_\chi - u_{fv}\|_\infty$, where u_{fv} is the numerical solution obtained by the finite volume method, and u_χ is the numerical solution from the truncation method. This means that as the size of the inviscid region is reduced, the error between the vector truncation method and the finite volume method is also reduced. Varying the value of σ, i.e. the size of the transition region, also has an effect on the error. Again it is observed that as the value of σ is increased the error also increases. However the error due to increasing σ does not appear to be as significant, and it depends to a certain extent on the chosen value of ϵ. For example, large ϵ together with large σ will produce a large error due to the viscous region being very small. However if ϵ is small then the effect of having a large σ is greatly reduced. It is clear that careful consideration is required if the best combination for these two parameters is to be chosen.

4 Navier-Stokes/Euler Coupling

With the vector truncation method in place it is possible to use this truncation technique as the basis for an adaptive zonal recognition procedure. The finite volume code has been further developed so that once identified the regions are decoupled and solved separately.

An Euler code has been incorporated into this single code framework to solve the large inviscid region. The Euler equations are written as,

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0, \tag{4.8}$$

where

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho v \end{pmatrix}, \qquad F = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \end{pmatrix}, \qquad G = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \end{pmatrix}. \tag{4.9}$$

For incompressible flow the energy equation can be replaced by the condition of constant total enthalpy

$$\frac{\gamma}{\gamma - 1}\frac{p}{\rho} + \frac{1}{2}(u^2 + v^2) = H_\infty. \tag{4.10}$$

The discretised Euler equations are rearranged so that grid points along a vertical column of the body fitted coordinate form a tridiagonal system. The whole region is solved column by column using a tridiagonal solver, which corresponds to a semi-implicit scheme. The algorithm is easy to implement and is sufficient at this stage to test the coupling procedure.

The algorithm for this single framework is shown in Figure 2. A finite volume Navier-Stokes code is run for a number of sweeps so that the solution develops just enough for the zonal boundaries to be identified. Decoupling then occurs with the initial solution being mapped onto the two subdomains. The existing finite volume Navier-Stokes code is coupled with a finite volume Euler code. Suitable Neumann and Dirichlet boundary conditions are imposed for the purpose of exchanging information between the two subdomains.

5 Numerical Results

Numerical tests have been carried out on a 1m flat plate problem with threshold parameters initially set at $\epsilon = 0.01$ and $\sigma = 0.0001$ (see Fig. 3). These results are promising although the coupling between viscous and inviscid regions still requires some adjustments. The internal interface between the two regions uses Dirichlet/Neumann boundary conditions for Navier-Stokes and Euler regions respectively. Other combinations have been tried but this appears to give the best results. The discrepancy in the Euler solution close to the viscous/inviscid interface is being investigated. This may be caused by the way in which the boundary conditions are applied.

6 Discussion and Conclusions

Viscous and inviscid regions of a flow domain can be identified by means of a vector truncation method within a finite volume Navier-Stokes framework. The regions are decoupled and solved iteratively within a single code. This is a great advantage as it saves computational time and overcomes the problems of more traditional coupling methods. Zones no longer need to be predefined and there is no need to choose boundary conditions for the overlapping zonal boundaries. Boundary values can be taken from the developing solution.

For greater computational speed the Euler/Navier-Stokes coupling may also be thought of as two codes working within a single framework. This enables the two codes to be run simultaneously on two processors with boundary information being exchanged between processors at regular intervals.

Increased accuracy in the viscous region is an important aspect. Greater accuracy in the boundary layer could be achieved by adapting the mesh in this region. The truncation method will be used for zonal recognition and mesh points will be drawn from the inviscid region into the viscous region. In this way greater accuracy can be obtained where the physics is most interesting without having to add points.

REFERENCES

[AC93] Arina R. and Canuto C. (1993) A self-adaptive domain decomposition for the viscous/inviscid coupling. I. Burgers equation. *Journal of Computional Physics* 105: 290+.

[AP93] Achdou Y. and Pironneau O. (1993) The χ-method for the Navier-Stokes equations. *IMA Journal of Numerical Analysis* 13: 537+.

[BCR89] Brezzi F., Canuto C., and Russo A. (1989) A self-adaptive formulation for the Euler/Naiver-Stokes coupling. *Computer Methods in Applied Mechanics and Engineering* 73: 317+.

[Kno86] Knott M. J. (1986) *Numerical Solutions of Two-Dimensional Unsteady Boundary Layers.* PhD thesis, University of London.

[LCP96] Lai C. H., Cuffe A. M., and Pericleous K. A. (1996) A domain decomposition algorithm for viscous/inviscid coupling. *Advances in Engineering Software* 26: 151+.

[Mar93] Margot X. M. (August 1993) *A Physically Guided Zonal Approach for Euler/Navier-Stokes Predictions of Aerofoil Flows.* PhD thesis, University of

London, Department of Mechanical Engineering, Imperial College.

[PR89] Perkins A. L. and Rodrigue G. (1989) A domain decomposition method for solving a two-dimensional viscous burgers' equation. *Applied Numerical Mathematics* 6: 329+.

[SH86] Schmatz M. A. and Hirschel E. H. (1986) Zonal solutions for airfoils using Euler, boundary-layer and Navier-Stokes equations. In *Applications of Computional Fluid Dynamics in Aeronautics, AGARD Conference proceedings, 412*.

Figure 2 Flow diagram

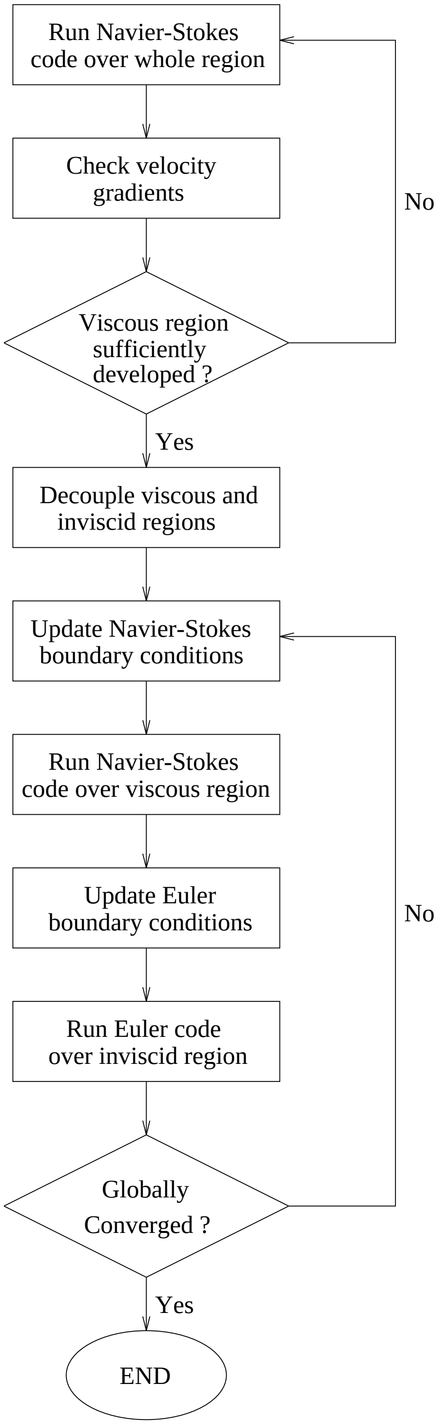
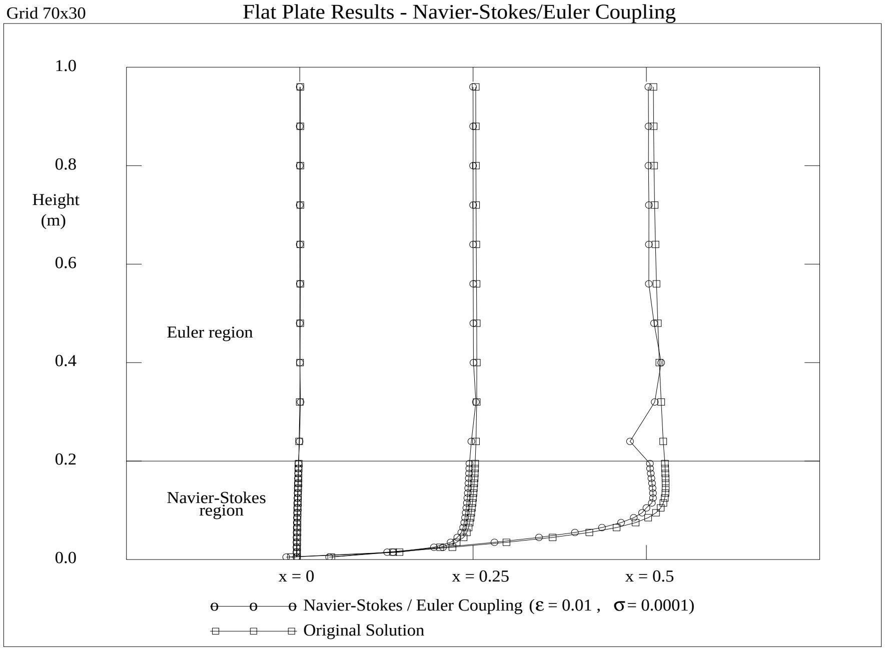

Run Navier-Stokes code over whole region
Check velocity gradients
Viscous region sufficiently developed ?
No
Yes
Decouple viscous and inviscid regions
Update Navier-Stokes boundary conditions
Run Navier-Stokes code over viscous region
Update Euler boundary conditions
Run Euler code over inviscid region
Globally Converged ?
No
Yes
END

Figure 3 Graph to show velocity profiles at 3 points along a 1m Flat Plate



82

A Characteristic Domain Decomposition Method for Modeling Flow in a Coastal Aquifer

Helge K. Dahle, Torbjørn O. Widnes Johansen, Tone Botnen, and Xue-Cheng Tai

1 Introduction

A characteristic domain splitting method is implemented for the concentration equation of a coastal aquifer with intrusion and discharge. In each timestep the concentration is advected along streamlines. The diffusion part is then solved using an overlapping domain decomposition technique. If the overlapping size is suitably chosen, no iterations are needed between the subdomain problems at each time level. If the diffusion parameter or the timestep is sufficiently small, only one or two elements of overlap is needed. For problems with large diffusion, or if we use large timesteps, a few iterations between the subproblems are needed to further reduce the domain decomposition error. Numerical results show the potential of this method for the ground water flow problem.

Salt water consists of one liquid phase composed of salt and water components. The mixing of salt and fresh water in coastal aquifers may be described by Darcy's law and conservation of mass. Here, a two-dimensional model of a coastal aquifer is considered [Bot93, SRMS92]. The x-direction is aligned with the main horizontal flow direction and z denotes the vertical direction pointing upwards from the bottom of the aquifer; see Figure 1. We shall assume that the aquifer is completely saturated by water, that the density of water depends linearly on salt concentration, i.e., $\rho = \rho_0(1 + \beta c)$, that the hydraulic conductivity tensor is isotropic and diagonal, $\mathbf{K} = K(\mathbf{x})\mathbf{I}$, and porosity $n = 0.3$. These assumptions leads to a somewhat simplified set of governing equations for $(\mathbf{x}, t) \in \Omega \times (0, T]$:

$$\mathbf{q} = -K(\nabla\phi + \beta c \nabla z), \tag{1.1}$$

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

$$-K\nabla^2\phi = \beta(K\frac{\partial c}{\partial z} - n\frac{Dc}{Dt}), \tag{1.2}$$

$$\frac{Dc}{Dt} - \nabla \cdot D_h \nabla c = 0. \tag{1.3}$$

Here $\mathbf{q} = n\mathbf{v}$ is the volumetric flow, $\mathbf{v}$ is the particle velocity, c is the concentration, $\phi = p/\rho_0 g + z$ is the fresh water head, D_h represents hydromechanical dispersion and $D/Dt = \partial/\partial t + \mathbf{v}\cdot\nabla$ denotes the convective derivative. According to site investigations, a good estimate of the dispersion term is $D_h \sim 10^{-5} m^2/s$, see [SRMS92]. However, we note that this term generally is given by a tensor which is difficult to model and may be impossible to measure by direct means, e.g., [BB90]. Hence, a reliable and efficient numerical model may be needed to estimate this term. A similar statement may be made about the conductivity. However, this term is somewhat easier to measure by direct means, and in Figure 1 data for the conductivity are given. The flow takes place in a two-dimensional region Ω bounded by a river, a channel, the ocean (fjord) and impervious rock. For simplicity, we assume stationary boundary conditions. The boundary conditions, being a mixture of Neumann and Dirichlet conditions, are shown in Figures 2 and 3. Initially, the aquifer is completely filled with salt water. Since the boundary conditions are stationary this leads to a stationary solution after some time ($\sim$ 200 days), describing a mixing zone between fresh and salt water. We do not intend to describe the effect of tidal water, seasonal changes or wells on this mixing zone. Typical length and time scales for this problem are $L = 100m$ and $T = 100$ days respectively. This gives a typical diffusion in the range $0.005 < D_h < 0.5$. For simplicity we keep the parameters in dimensional form, although concentration is scaled to vary between 0 and 1 in the figures. The main aim of the present paper is to report numerical experiments when the solution method for the advection-dispersion equation is a part of a complicated, coupled, nonlinear system of partial differential equations. This is done in section 1.3. The rest of the paper describes how equation (1.3) is solved, by extending and modifying the algorithm given in [TJDE96].

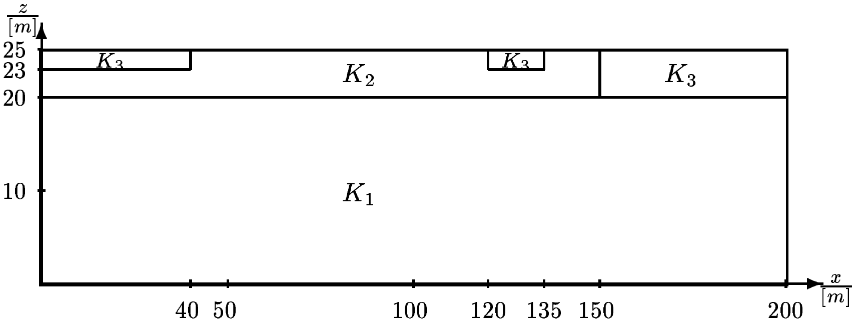
2 Algorithms

Equations (1.1)-(1.3) are solved using a sequential time-marching procedure: At each time level t^n the pressure/velocity equations (1.1), (1.2) are solved using the previous known concentration values. The concentration distribution is then updated using the new velocity field. In this way the issue of determining pressure/velocity is decoupled from the problem of finding a concentration distribution. In particular, the pressure velocity equations are solved by a control volume technique, see [DEEa90]. Here, we focus on the transport equation (1.3), describing the mixing process. Thus, from now on the velocity field is assumed to be a known function of space and time.

Discretization

The concentration equation will be solved by the Modified Method of Characteristics (MMOC), see [DES92, DR82]. This choice is important in two ways. First, it gives

Figure 1 Hydraulic conductivity of the modeled profile.
$K_1 = 6.5 \cdot 10^{-4}\, m/s$, $K_2 = 6.5 \cdot 10^{-3}\, m/s$, and $K_3 = 0.1\, m/s$.



an accurate solution even for large timesteps since the coefficients of the time-truncation error only depends on higher order derivatives along the (approximate) characteristics, see [DR82]. Secondly, it gives very accurate internal boundary values for the subdomains, to be used in the domain decomposition method. To be more precise, let $S_h(\Omega) \subset H^1(\Omega)$ be the space of piecwise bilinear functions on a rectangular discretization of Ω. This defines a finite element discretization $\Omega = \cup_{e \in \mathcal{T}_h} e$. Let V^h be the subset of $S_h(\Omega)$ satisfying the given Dirichlet conditions and $S_h^0(\Omega)$ be the subspace of functions which are zero at the Dirichlet part of the boundary. The MMOC approximation may then be written: For $n = 1, 2, ...$, find $c^n \in V_h$ such that

$$(c^n, v) + (\Delta t D_h \nabla c^n, \nabla v) = (\bar{c}^n, v) + \mathrm{B.}T., \quad \forall v \in S_h^0(\Omega). \tag{2.4}$$

Here $\Delta t = t^n - t^{n-1}$ is the timestep, $(\cdot, \cdot)$ is the usual L_2-inner product on Ω and B.T. denotes boundary terms. The characteristic solution $\bar{c}^n \in V_h$ is obtained by tracking particle trajectories backwards in time from each node $\mathbf{x}_i$, i.e., $\bar{c}^n(\mathbf{x}_i) = c^{n-1}(\bar{\mathbf{x}}(\mathbf{x}_i, t^{n-1}))$, where

$$\frac{d\bar{\mathbf{x}}}{d\tau} = \mathbf{v}(\bar{\mathbf{x}}(\mathbf{x}_i, \tau), \tau) \ \ and \ \ \bar{\mathbf{x}}(\mathbf{x}_i, t^n) = \mathbf{x}_i, \quad \tau \in [t^{n-1}, t^n]. \tag{2.5}$$

In the present work $\mathbf{v}(\mathbf{x}, t)$ is approximated by $\mathbf{v}(\mathbf{x}, t^n)$. If tidal effects, etc., are important linear interpolation between successive time levels may be necessary. In the numerical experiments presented here, Equation (2.5) is solved by a one-point approximation using the tangent of the velocity field at the node in consideration. Note that this can be done in parallel and gives a fast and robust method. More accurate methods are, e.g., analytical integration or Runge-Kutta methods. The main difficulty with the MMOC is when a characteristic traced backward in time crosses the physical boundary; see [WDE$^+$96] and references therin. This may happen at an inflow or a noflow boundary. In the present case, inflow boundaries are easy to treat since concentration values are specified to be either 0 or 1. Crossing a noflow boundary, due to inexact tracing, is treated by projecting the characteristic back onto the boundary.

Figure 2 Boundary conditions for the fresh water head equation.

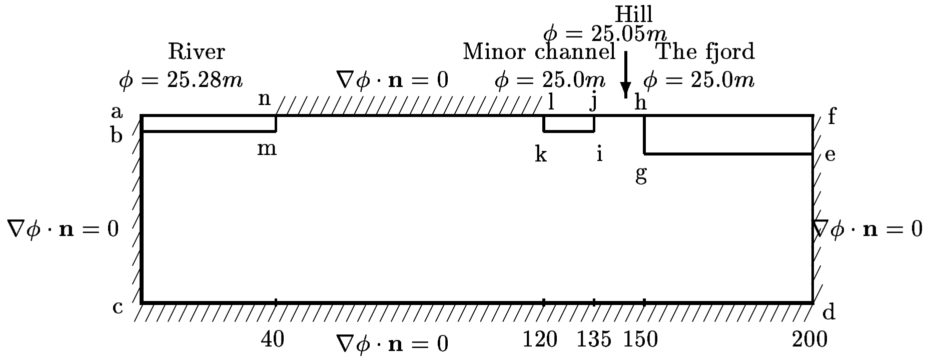
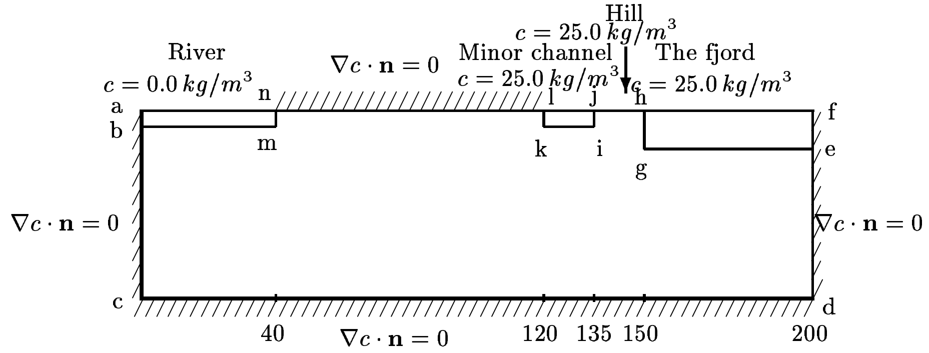

Hill
$\phi = 25.05m$
River
$\phi = 25.28m$
$\nabla\phi \cdot \mathbf{n} = 0$
Minor channel
$\phi = 25.0m$
The fjord
$\phi = 25.0m$
a
b
c
d
e
f
g
h
i
j
k
l
m
n
$\nabla\phi \cdot \mathbf{n} = 0$
$\nabla\phi \cdot \mathbf{n} = 0$
40
$\nabla\phi \cdot \mathbf{n} = 0$
120
135
150
200

Figure 3 Boundary conditions for the concentration equation.



Characteristic Domain Decomposition

Let Ω be decomposed into M nonoverlapping subdomains Ω_i. For simplicity Ω is only subdivided in the horizontal direction. To each Ω_i we associate an enlarged subdomain

$$\Omega_i^\delta = \{e \in \mathcal{T}_h|\ dist(e, \Omega_i) \leq \delta\},$$

which forms an overlapping domain decomposition of Ω with overlapping size δ. In practice δ is measured in terms of the number of elements that Ω_i extends into its neighbors. On each subdomain, solve the following problem: Find $c_i^n \in V_i^h$ such that $c_i^n = \bar{c}^n$ on $\partial\Omega_i^\delta \backslash \partial\Omega$ and

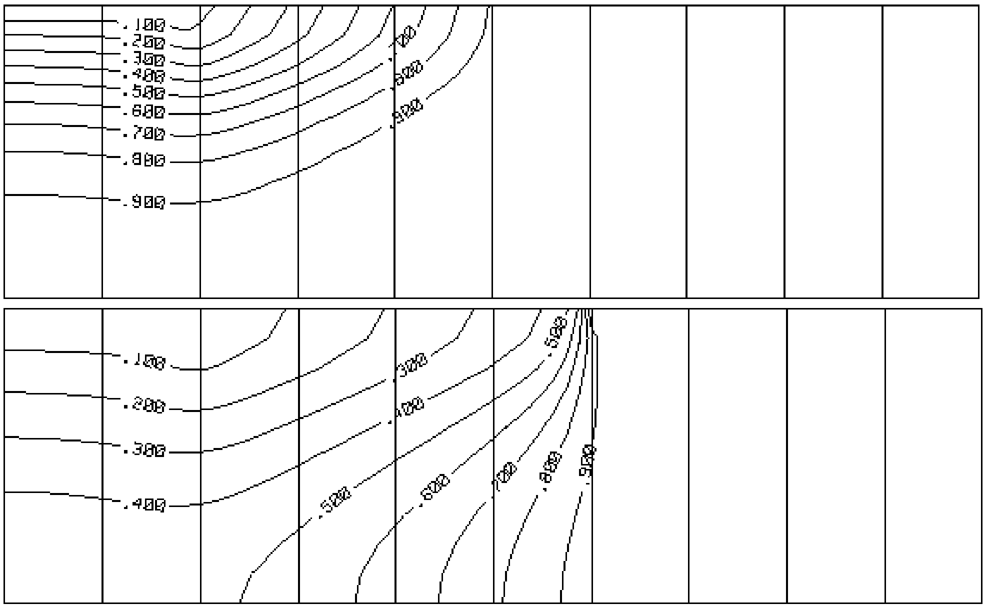
$$(c_i^n, v) + (\Delta t D_h \nabla c_i^n, \nabla v) = (\bar{c_i}^n, v) + \mathrm{B}.T., \quad \forall v \in S_h^0(\Omega_i^\delta). \tag{2.6}$$

Here, $S_h^0(\Omega_i^\delta)$ is the finite element subspace of functions which are zero at the Dirichlet part of $\partial\Omega_i^\delta$ and V_i^h is the restriction of V^h to Ω_i^δ.

Note that the subproblems can be solved in parallel. After solving the subproblems, we assemble a global solution from the subdomain solutions. The following algorithm may now be stated, see also [TJDE96]:

Algorithm *For each time level t^n:*

Figure 4 10 (top) and 50 (bottom) timesteps



1. Solve (2.5) to obtain $\bar{c}^n \in V^h$;

2. Solve (2.6) on each subdomain Ω_i^δ to get c_i^n;

3. From the patchwise solution c_i^n, a global solution

$$c^n = \mathcal{C}\left(\{c_i^n\}_{i=1}^M\right) \in V^h$$

is constructed such that

$$\| c^n \|_{L^2(\Omega)} \leq \sum \| c_i^n \|_{L^2(\Omega_i)}; \tag{2.7}$$

4. If $t^n < T$, got to next time level;

We may iterate between Step 2 and 3 to further improve the solution. Step 3 is achieved in practice by a cutting and averaging technique [BLR92, TJDE96], i.e. we set the value of c^n by

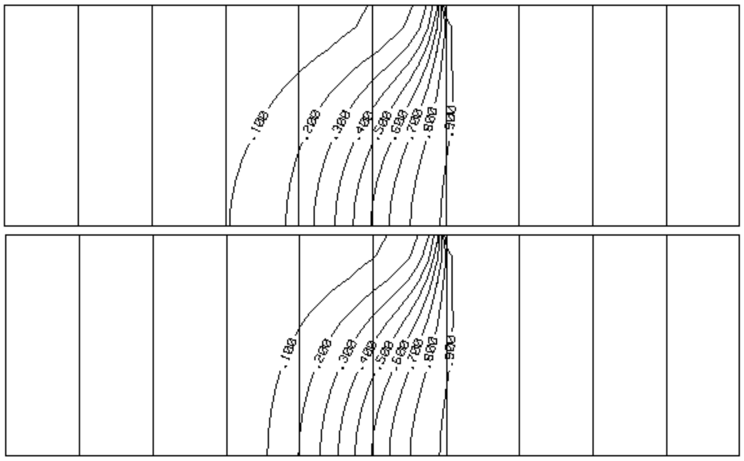
$$c^n(x_k) = \begin{cases} c_i^n(x_k), & \text{if } x_k \text{ is an inner node of } \Omega_i, \\ \frac{1}{2}(c_i^n(x_k) + c_j^n(x_k)), & \text{if } x_k \text{ is a node on the interface between of } \Omega_i \text{ and } \Omega_j. \end{cases}$$

It was proved in [TDE] that if

$$\delta > c_0 \max(\sqrt{\epsilon \Delta t}, h)|ln\Delta t|,$$

where c_0 is a constant associated with the finite element mesh and h is the mesh size, then the computed solution c^n is of first order of convergence with respect to Δt and second order of convergence with respect to h.

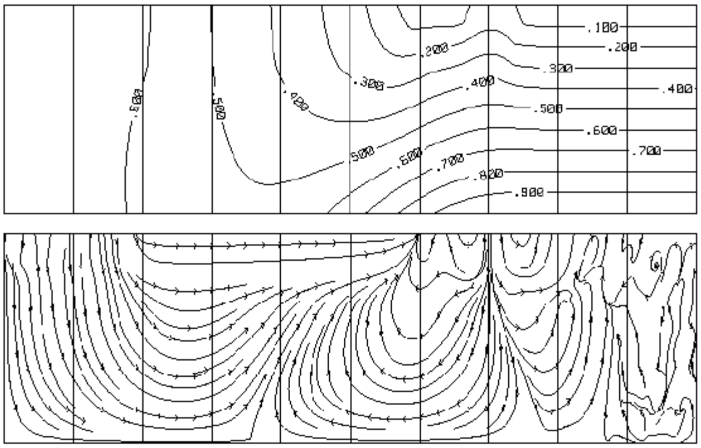
Figure 5 Concentration distribution. $D_h = 5 \cdot 10^{-5} m^2/s$.



3 Numerical Experiments

In the experiments performed, the problem is solved on a uniform grid with 81×11 grid lines, and the domain is divided into 8 equally sized subdomains. The timestep is fixed to be 24 hours. Initially the aquifer is filled with salt water ($c = 1$) and the boundary conditions are prescribed as shown in Figure 3. Fresh water ($c = 0$), then infiltrates from the river (upper right corner). In Figure 5 contour plots of the concentration distribution is shown after 10, 50, 150 and 250 timesteps for $D_h = 5 \cdot 10^{-5} m^2/s$. Figure 6 show the related pressure distribution and streamlines after 250 timesteps. At this point the problem has reached a stationary solution which is independent of the initial conditions. A reference (global) solution is computed on the same grid without domain decomposition. Figures 7 - 8 compare the global solution with the domain decomposition solution in a discrete L_2-norm, i.e. the L_2-error. Figure 7 show the error as a function of number of overlapping elements (no iterations) for a fairly small diffusion ($D_h = 5 \cdot 10^{-5} m^2/s$). The error decays as expected and only 2-3 elements were necessary to obtain an accurate solution. This is due to the fact that the characteristic solution is nearly exact for small diffusion. On the other hand, the discontinuity at the inflow boundary produced a transient phase with big errors. After two timesteps the error could not be forced to zero by a reasonable increase of the number of overlapping elements, as shown in Figure 7. This problem is explained by the fact that a small change in the velocity field, caused by small differences in concentration values, produces big differences in the concentration values in vicinity of a discontinuous infiltration front. The problem disappeared after ~ 50 timesteps. Figure 8 shows the error for a large diffusion ($D_h = 3 \cdot 10^{-4} m^2/s$). For this problem a combination of 2-3 elements of overlap and 1-2 iterations between the subdomains at each time level reduced the error to an accepted level. We did not observe any difficulties in this case, since the infiltration front was immediately smeared by the

Figure 6 Fresh water head and streamlines after 250 timesteps.



diffusion term.

4 Conclusions

The combination of sequential timestepping and characteristic domain decomposition as described above, is easy to implement and gives fast and robust methods. The numerical experiments performed here show results that are expected from analysis for a single linear advection diffusion equation. In fact, for small diffusion 2-3 elements of overlap without iterations seems to be sufficient, for a larger diffusion 2-3 elements of overlap combined with 1-2 iterations between the subdomains at each time level are needed. However, more experiments have to be done and the algorithms should be implemented on a parallel machine for measuring speed up times.

Acknowledgement

We would like to thank O. Soldal and N. Rye for providing data and valuable discussions on the Geological model, and O. Sævareid for providing a first version of the Pressure/Velocity code. The work was supported by the University of Bergen and by VISTA, a research cooperation between the Norwegian Academy of Science and Letters and Den norske stats oljeselskap AS (Statoil).

REFERENCES

[BB90] Bear J. and Bachmat Y. (1990) *Introduction to Modeling of Transport Phenomena in Porous Media.* Kluwer Academic Publishers.

[BLR92] Blum H., Lisky S., and Rannacher R. (1992) A domain splitting algorithm for parabolic problems. *Computing* 49: 11–23.

[Bot93] Botnen T. H. (1993) Mathematical and numerical modeling of a coastal aquifer. Master's thesis, University of Bergen, Department of Mathematics.

[DEEa90] Dahle H. K., Espedal M. S., Ewing R. E., and areid O. S. (1990) Characteristic adaptive subdomain methods for reservoir flow problems. *Numerical Methods for Partial Differential Equations* 6: 279–309.

[DES92] Dahle H. K., Espedal M. S., and Sævareid O. (1992) Characteristic, local grid refinement techniques for reservoir flow problems. *International Journal for Numerical Methods in Enginering* 34: 1051–1069.

[DR82] Douglas J. and Russell T. F. (October 1982) Numerical methods for convection-dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures. *SIAM J. Numer. Anal.* 19(5): 871–885.

[SRMS92] Soldal O., Rye N., Mauring E., and Sæther O. M. (1992) Saline groundwater extraction from the fjord delta aquifer, sunndalsøra, møre og romsdal, norway. *NGU Bulletin* 422: 37–46.

[TDE] Tai X.-C., Dahle H. K., and Espedal M.A characteristic domain splitting method for time dependent convection-diffusion problems. (In preparation).

[TJDE96] Tai X.-C., Johansen T., Dahle H. K., and Espedal M. (1996) A characteristic domain splitting method. In Glowinski R., Périaux J., Shi Z.-C., and Widlund O. B. (eds) *Proc. Eighth Int. Conf. on Domain Decomposition Meths.* Wiley and Sons, Chichester.

[WDE$^+$96] Wang H., Dahle H. K., Espedal M. S., Ewing R. E., Sharpley R. C., and Man S. (1996) An ELLAM scheme for advection-dispersion equations in two dimensions. Preprint.

Figure 7 Error after 250 timesteps (top figure) and after 2 timesteps (bottom figure) as a function of the number of overlapping elements. $D_h = 5 \cdot 10^{-5} m^2/s$.

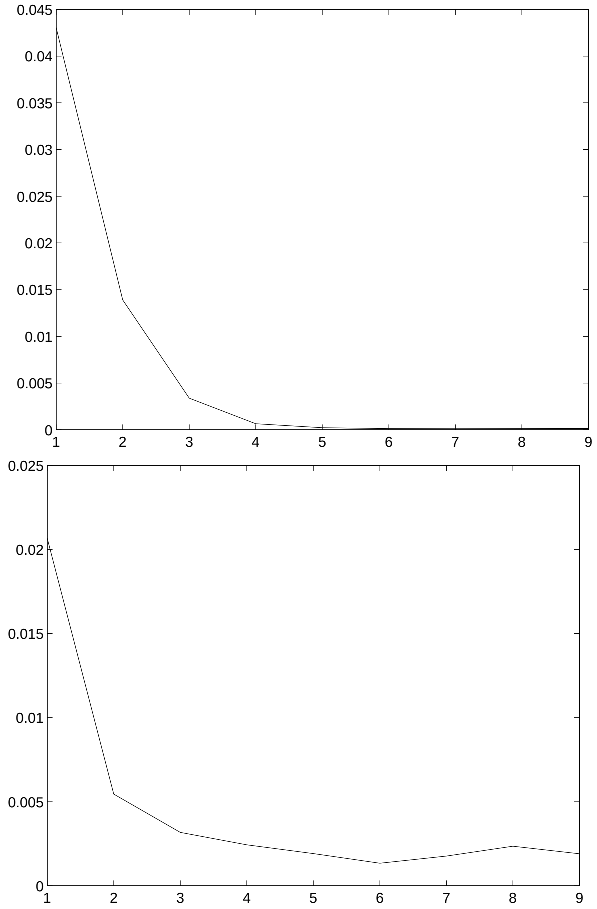
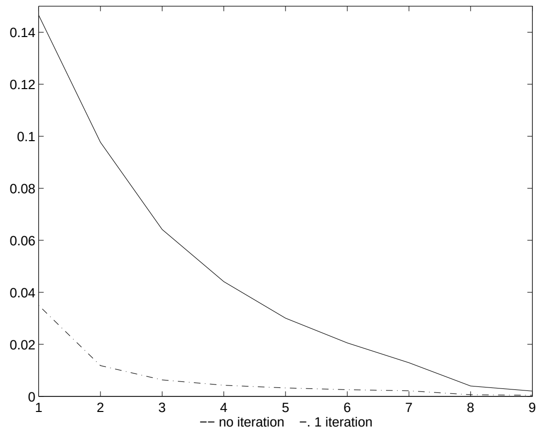


Figure 8 Error after 250 timesteps as a function of the number of overlapping elements. $D_h = 3 \cdot 10^{-4} m^2/s$.



83

Applications of Dual Schur Complement Preconditioning to Problems in Computational Fluid Dynamics and Computational Electro-Magnetics

Quang Vinh Dinh and Thierry Fanion

1 Introduction

Domain-based parallel implementation of numerical codes using unstructured grids have been very successful for codes based on explicit integration schemes. For implicit schemes, which require successive linear solves, there is still room for improvement and research toward an efficient linear solver based on domain partitioning. In particular, since an efficient solution of the overall nonlinear problem does not require each successive linear problem to be solved to maximum accuracy, iterative methods are usually preferred.

In [Ven94], linear systems are solved by a preconditioned iterative method with a block diagonal preconditioner on interfaces and involving, within each subdomain, an incomplete factorization corresponding to a fixed sparsity pattern. While the local and parallel solves are efficient since their cost are linear in size, the convergence of the outer method degrades when the number of subdomains is high. A coarse grid solver has also been proposed in [Ven94] to alleviate this problem, at the cost of introducing a complex agglomeration procedure.

In [FMR94], a dual Schur complement method is presented for linear problems in elasticity: it is shown that the dual version of the method is preferable from a spectral convergence theory point of view. Indeed, the number of "outer" Schur iterations does not depend much on the number of subdomains into which the initial mesh has been divided. But this remarkable result is achieved with the use of direct solvers in each subdomain. Reusing previous right-hand sides at the "outer" level in reconjugation techniques also proves to be efficient: this is shown in [Rou94] for nonlinear problems in elasticity.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Following these lines, we would like to find a domain-partitioned linear solver which is suitable for applications in Computational Fluid Dynamics (CFD) and Computational Electro-Magnetics (CEM). We begin by describing our CFD solver; from its parallel implementation on distributed memory machines, we infer the desired characteristics of our parallel linear solver. We then present three solvers based on dual Schur complement methods and discuss their merits on representative problems. A more realistic three-dimensional result is then given to support our discussion. Lastly, we present some future work, applying these techniques to CEM problems.

2 VIRGINI: a CFD Solver for Low and High-speed Aircraft Design

VIRGINI is a two/three-dimensional Navier-Stokes solver developed at Dassault-Aviation for the last 10 years. It is extensively used for the simulation of viscous flows including modelization of turbulence phenomena and nonequilibrium air. We refer to [CMR94] and the bibliography therein for a complete description of its ingredients, which we would like now to review briefly.

Governing Equations

For the sake of simplicity, we restrict ourselves to viscous turbulent flows. Let ρ, $\boldsymbol{u}$, and E denote respectively the density, the velocity, and the total energy per unit mass of fluid. The mass-averaged Navier-Stokes equations for a compressible viscous fluid read as conservation laws in flow domain Ω for:

$$\begin{array}{lrcll}
\text{mass} & \dfrac{\partial\rho}{\partial t} + \nabla\cdot(\rho\boldsymbol{u}) & = & 0, & (2.1)\\
\text{momentum} & \dfrac{\partial\rho\boldsymbol{u}}{\partial t} + \nabla\cdot(\rho\boldsymbol{u}\otimes\boldsymbol{u}) & = & \nabla\cdot\boldsymbol{\sigma}, & (2.2)\\
\text{energy} & \dfrac{\partial\rho E}{\partial t} + \nabla\cdot(\rho E\boldsymbol{u}) & = & \nabla\cdot(\boldsymbol{\sigma}\boldsymbol{u}) - \nabla\cdot\boldsymbol{q}, & (2.3)
\end{array}$$

where $\boldsymbol{\sigma}$ is the Cauchy-Reynolds shear stress tensor and $\boldsymbol{q}$ is the heat-flux vector. Appropriate boundary conditions, usually of the no-slip type, are enforced on $\partial\Omega$. Using the above set of equations to describe the mean flow, we rely on a classical Boussinesq hypothesis and the concept of eddy viscosity to make the required turbulence closure assumptions, which lead us to the following definitions for the stress tensor

$$\boldsymbol{\sigma} = (\mu^{\text{visc}} + \mu_t^{\text{visc}})\{\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^t - \frac{2}{3}\nabla\cdot\boldsymbol{u}1\} - (p + \frac{2}{3}\rho k)1,$$

the total energy

$$E = e + \frac{1}{2}|\boldsymbol{u}|^2 + k,$$

and the heat-flux vector,

$$\boldsymbol{q} = -(\mu^{\text{visc}}\frac{\gamma}{P_r} + \mu_t^{\text{visc}}\frac{\gamma}{P_{rt}})\nabla e.$$

Here μ^{visc} is the molecular viscosity, μ_t^{visc} is the eddy viscosity, 1 is the identity tensor, k is the turbulent kinetic energy, $\gamma = c_p/c_v$ is the ratio of the fluid specific heats. The laminar Prandtl number is taken as $P_r = 0.72$ and the turbulent Prandtl number is taken as $P_{rt} = 0.9$. Internal energy e is defined by $e = c_v T$ and pressure p is calculated from the thermodynamic state equation $p = p(\rho, T)$.

The turbulence model used belongs to the $k-\epsilon$ family (see [CMR94]) and introduces two extra equations:

$$\begin{aligned} \rho\frac{\partial k}{\partial t} + \rho\boldsymbol{u}\cdot\nabla k - \nabla\cdot((\mu^{\text{visc}} + \frac{\mu_t^{\text{visc}}}{\sigma_k})\nabla k) &= H_k(k, \epsilon, \rho, \boldsymbol{u}, \mu_t^{\text{visc}}) &(2.4)\\ \rho\frac{\partial \epsilon}{\partial t} + \rho\boldsymbol{u}\cdot\nabla \epsilon - \nabla\cdot((\mu^{\text{visc}} + \frac{\mu_t^{\text{visc}}}{\sigma_\epsilon})\nabla \epsilon) &= H_\epsilon(k, \epsilon, \rho, \boldsymbol{u}, \mu_t^{\text{visc}}) &(2.5)\end{aligned}$$

which are solved for k and ϵ. The eddy viscosity is then defined as:

$$\mu_t^{\text{visc}} = \rho C_\mu \frac{k^2}{\epsilon}.$$

Here $\sigma_k, \sigma_\epsilon, C_\mu$ are modelling constants and H_k, H_ϵ are source terms.

Numerical Approximation

Different numerical approximations are used for the two systems of equations above.

For system (2.4-2.5), corresponding to the turbulence model, the positivity of k and ϵ is achieved by the combination of two main features (see [CMR94]): the use of a monotone advective finite-volume scheme and the time discretization of the source terms. This is done via a semi-implicit time-marching algorithm, leading to two decoupled linear systems to be solved at each time iteration:

$$\rho\frac{\Delta i}{\Delta t} + \rho\boldsymbol{u}\cdot\nabla i - \nabla\cdot((\mu^{\text{visc}} + \frac{\mu_t^{\text{visc}}}{\sigma_i})\nabla i) = H_i, \qquad (2.6)$$

where $i = k$ or ϵ, and Δi (resp. Δt) is the variable (resp. time) increment.

In system (2.1-2.3) representing the mean flow equations (see [CMR94]), a Galerkin/least-squares finite element formulation is applied to the compressible Navier-Stokes equations which has been rewritten in the form of a symmetric advective-diffusive system in terms of entropy variables $\boldsymbol{V}^T = \partial\mathcal{H}/\partial\boldsymbol{U}$, where $\boldsymbol{U}^T = \{\rho, \rho\boldsymbol{u}, \rho E\}$ are the conservative variables and $\mathcal{H}(\boldsymbol{U}) = -\rho s$ is the generalized entropy function, with s being the physical entropy per unit mass. A fully implicit time-marching procedure is used, so that a nonlinear problem is solved at each time step. A linearization through a truncated Taylor series expansion is then performed, leading to the following linear system to be solved for variable increment $\Delta\boldsymbol{V}$:

$$\widetilde{\boldsymbol{A}}_0\frac{\Delta\boldsymbol{V}}{\Delta t} + \widetilde{\boldsymbol{A}}\cdot\nabla\Delta\boldsymbol{V} = \nabla\cdot(\widetilde{\boldsymbol{K}}\nabla\Delta\boldsymbol{V}) + \mathcal{F} \qquad (2.7)$$

in which

$\widetilde{\boldsymbol{A}}_0$ is symmetric and positive definite,

$\widetilde{\boldsymbol{A}} = \{\widetilde{\boldsymbol{A}}_i\}, 1 \leq i \leq 5$ with $\widetilde{\boldsymbol{A}}_i$ symmetric,

$\widetilde{\boldsymbol{K}}$ is symmetric and positive semi-definite,

$\mathcal{F}$ is the current right-hand side.

Finally, the discretized mean flow equations and the turbulence equations are coupled through a splitting method. At a current time step, we solve the Navier-Stokes equations using turbulence data evaluated at the previous time while the turbulence equations are solved using the flow variables computed at the previous time.

All space discretizations are done on unstructured meshes with piecewise linear interpolation on tetrahedra. Since we are using VIRGINI to get steady-state solutions, time accuracy is not mandatory. Thus, the above linear systems need not be solved to maximum accuracy and our preferred linear solver is an iterative method, namely GMRES (see [BBC$^+$94]), with diagonal preconditioning for (2.6) and nodal block-diagonal preconditioning for (2.7).

Computer Implementation

An iterative linear solver like GMRES requires only two types of operations (see [BBC$^+$94]): vector inner-product and matrix-vector product. This allows us to implement in VIRGINI the so-called "matrix-free" procedure, that is matrices for systems (2.6) and (2.7) are never stored, only procedures to compute the corresponding matrix-vector products are created. This feature, along with the usage of diagonal preconditioning, give an "explicit-like" behavior to the code, which facilitates vectorization via coloring techniques and parallelization via domain partitioning.

VIRGINI has been ported on various platforms: IBM ES-9000, Convex C2-C3, NEC-SX3, Intel iPSC-860, IBM SP2, workstations (IBM RS6000, SGI), etc. The memory requirement is about 2.7KB per mesh node, which means that, on our 16-processor IBM SP2 with "thin" nodes (i.e., with 128MB of local memory), we can accommodate a "maximum-size" mesh of 750,000 nodes. We give some typical convergence data in the following simplified flowchart of VIRGINI:

```
Begin time-marching loop

step 1: Solve (2.7) with previous turbulence data
    → convergence level required: $10^{-1}$ to $10^{-3}$
    → number of GMRES iterations: at most 10

step 2: Solve (2.6) with previous flow data
    → convergence level required: $10^{-3}$ to $10^{-5}$
    → number of GMRES iterations: at most 20

End time-marching loop if nonlinear residual is small enough
```

Improving Performance

We would like to improve the convergence of the linear solvers while retaining, as much as possible, the nice features of VIRGINI.

One usual way is to replace the diagonal preconditioner by a more elaborate one built from an incomplete LDU factorization of the linear operators. This preconditioner introduces some extra memory and computational burdens which must be carefully evaluated.

For system (2.7), preliminary tests have shown that it requires too much extra computation and, in particular too much extra memory, which cannot be accounted for in light of the low convergence requirement. The situation is different for system (2.6) which must be solved more accurately. But then, what becomes of the incomplete LDU preconditioner in a parallel framework involving domain partitioning?

An "ad hoc" local incomplete LDU preconditioner has been proposed in [Ven94], where Dirichlet type conditions are imposed on interfaces, allowing the decoupling of the original operator into local ones. This has been shown to work well for Euler solvers, but the convergence degrades as the number of subdomains increases.

We would like now to derive, in the framework of dual Schur complement methods, a parallel solver for system (2.6) based on local incomplete LDU factorization, which, ideally, should have the following characteristics:

- Convergence rate nearly independent of the number of subdomains, even at low levels of convergence
- Local solvers, the costs of which, both in computer time and in memory requirement, are linear in problem size.
- Improvement upon a global GMRES solver with diagonal preconditioning.

In what follows, incomplete LDU factorization, iLDU in short, will be understood to be with the sparsity pattern of the original matrix.

3 Dual Schur Complement Solvers

In this section, we first present three parallel solvers based on dual Schur complement methods and discuss their merits on two model problems which are representative of VIRGINI:

problem P1) two-dimensional flow around a NACA0012 airfoil at free-stream conditions: Mach number = 0.799, incidence = 2.26°, Reynolds number = 9×10^6. Mesh sizes are: 8008 nodes and 15794 triangles. The converged solution needs about 1000 time steps. As a model problem, we pick out the linear problem corresponding to system (2.6) at the 700th time step, where the flow is fully developed.

problem P2) three-dimensional flow over a blunt body at free-stream conditions: Mach number = 3.0, no incidence, Reynolds number = 10^6. Mesh sizes are: 3506 nodes and 13280 tetrahedra. We have chosen the linear turbulence problem at the 10th time-step, at the beginning of the convergence which takes about 200 time steps.

Secondly, a more realistic test case is presented in which the "best" solver is implemented.

Solver 0: Direct Local Solver

In [FMR94], a dual Schur complement method, named FETI (for Finite Element Tearing and Interconnecting), is presented for elliptic problems in elasticity. We briefly recall the ingredients of FETI, adapted to system (2.6).

In flow domain Ω discretized by an unstructured mesh (identified in the sequel with the flow domain), let A be the matrix of system (2.6), f the right-hand side vector and u the vector of unknowns. We partition Ω into N_s subdomains $\{\Omega^s\}_{1\leq s\leq N_s}$, and we denote by A^s, f^s, u^s and B^s, respectively, the subdomain discretization of operator A, the subdomain right-hand side and unknowns, and the signed matrix with entries $-1, 0, +1$ describing the subdomain interconnectivity. The original problem $Au = f$ is shown to be equivalent to the following one:

$$\begin{aligned} \forall s, 1 \leq s \leq N_s, \quad A^s u^s + {B^s}^T \lambda &= f^s && (3.8) \\ \sum_{s=1}^{N_s} B^s u^s &= 0, && (3.9) \end{aligned}$$

where λ is the Lagrange multiplier for constraint (3.9). As in the original FETI algorithm, the local systems (3.8) are solved by a direct method and there is an "outer" iterative procedure to compute λ: since A is nonsymmetric, we have used a GCR algorithm (see [BBC$^+$94]).

We have implemented this solver for problem P1 with different mesh partitions.

Table 1 CPU performance for solver 0

Number of subdomains	1	4	8	16	32
ideal iLDU	1.0	0.25000	0.12500	0.06250	0.03125
Solver 0	not rel.	0.03900	0.01400	0.01620	0.01740

The results are shown in table 1, where we have taken as CPU time unit, the CPU time taken by a global GMRES algorithm with iLDU preconditioning, converged to 10^{-16}, on the whole mesh Ω. The other entries in the "ideal iLDU" line have been computed assuming that this GMRES algorithm has been somehow "perfectly" parallelized. Values in the "Solver 0" line are actual measures.

For different values of N^s, solver 0 performs better than the "ideal" iLDU preconditioner, although for $N^s \geq 8$ there is a performance stagnation due to the importance of communication versus computation. However, these results were obtained for a level of convergence up to machine precision, which is required for direct methods. Moreover, these direct methods entail extra burdens at the subdomain level: $O(n_s^{2.5})$ in computation and $O(n_s^{1.5})$ in memory, where n_s is the size of the local matrix A^s.

Solver 1: Iterative Local Solver

To make the extra work at the subdomain level, *linear* in size, one simple idea is to replace the local direct solvers by local iterative solvers. We have done this in solver 1, where the local solver is a GMRES algorithm with iLDU preconditioning. But now, we have to monitor two levels of convergence: ε_{glo} for the "outer" GCR algorithm and ε_{loc} for the local GMRES algorithm.

For problem P2, we have made a study of the relationship between ε_{glo} and ε_{loc} for two mesh partitions.

Table 2 Level of convergence ("outer" iterations count) for solver 1

2 subdomains	$\varepsilon_{glo} = 10^{-3}$	$\varepsilon_{glo} = 10^{-5}$	$\varepsilon_{glo} = 10^{-10}$
$\varepsilon_{loc} = 10^{-3}$	0.80×10^{-3}(8)	0.68×10^{-3}(19)	0.68×10^{-3} (44)
$\varepsilon_{loc} = 10^{-5}$	0.39×10^{-3}(8)	0.33×10^{-5}(19)	0.80×10^{-6} (44)
$\varepsilon_{loc} = 10^{-10}$	0.39×10^{-3}(8)	0.32×10^{-5}(19)	0.23×10^{-10}(44)

4 subdomains	$\varepsilon_{glo} = 10^{-3}$	$\varepsilon_{glo} = 10^{-5}$	$\varepsilon_{glo} = 10^{-10}$
$\varepsilon_{loc} = 10^{-3}$	0.72×10^{-3}(17)	0.66×10^{-3}(31)	0.66×10^{-3} (61)
$\varepsilon_{loc} = 10^{-5}$	0.29×10^{-3}(17)	0.71×10^{-5}(31)	0.63×10^{-6} (61)
$\varepsilon_{loc} = 10^{-10}$	0.29×10^{-3}(17)	0.35×10^{-5}(31)	0.42×10^{-10}(61)

The results are shown in table 2. As expected, it does not pay to converge more, at the subdomain level, than to the level of convergence fixed for the "outer" GCR algorithm: indeed, one should take $\varepsilon_{loc} = \varepsilon_{glo}$. Another interesting remark is that the higher this level of convergence is, the weaker the dependence of the rate of convergence on the number of subdomains, in a relative sense.

Solver 2: Direct Local Solver for Approximate Operator

In solver 1, we have seen that the scalability in the number of subdomains is dependent on the level of convergence required, local as well as global. This constraint is removed if, as in solver 0, the local solver is direct, but we have seen that this incurs too high a memory requirement. On the other hand, in the dual Schur formulation (3.8-3.9), the global operator is defined solely by its local representation. From the local incomplete LDU factorizations:

$$\forall s, 1 \le s \le N_s, \quad A^s \approx \widetilde{L}^s \widetilde{D}^s \widetilde{U}^s \quad = \quad \widetilde{A}^s, \tag{3.10}$$

we can define an operator $\widetilde{A}$ from its local contributions $\widetilde{A}^s$. Since the incomplete factorizations are done with the same sparsity pattern as for A^s, the extra work to compute and store $\widetilde{A}$ is linear in n_s and the local solvers are direct. Thus the application of solver 0 to $\widetilde{A}$ will result in a parallel solver having all the required characteristics.

If $\widetilde{A}$ is a good "approximation" to A, we can now propose solver 2 which has the following ingredients:

- global iterative algorithm: GMRES with level of convergence ε_{glo}
- global preconditioner: solver 0 applied to $\widetilde{A}$ with the same level of convergence

Table 3 Level of convergence ("outer" iteration count) for solver 2

	$\varepsilon_{glo} = 10^{-3}$	$\varepsilon_{glo} = 10^{-5}$	$\varepsilon_{glo} = 10^{-10}$
2 subdomains	$0.57 \times 10^{-4}(9)$	$0.44 \times 10^{-6}(12)$	$0.36 \times 10^{-11}(21)$
4 subdomains	$0.44 \times 10^{-4}(9)$	$0.41 \times 10^{-6}(12)$	$0.36 \times 10^{-11}(21)$

We have implemented solver 2 for problem 2 and results shown in table 3 suggest that we have come up with a good solution since the "outer" iteration count does not vary when we go from 2 to 4 subdomains.

A Realistic Test Case

To support our discussion, we have run VIRGINI, with solver 2 implemented for system (2.6), on a more realistic flow simulation: a low-speed flow around the forebody of a military aircraft, namely a Mirage 2000, at high angle of attack.

Free stream conditions were: Mach number = 0.2, incidence = 50°, altitude = 3000 meters. Mesh sizes were around 50,000 nodes and 275,000 elements.

We have made a thorough comparison, during the first 100 time-steps, between this version of VIRGINI and the original version. The gains, for a convergence level for system (2.6) fixed at 10^{-5}, were:

in iteration count $\approx 60\%$

in CPU time $\approx 4\%$.

The smaller gain in CPU time can be accounted for by the extra local incomplete factorizations done at each time step. Different operator "freezing" strategies should be used here to alleviate this problem.

The complete convergence for this simulation needed about $2,000$ iterations and over 33 hours on a IBM SP2 with 4 (thin) processors.

4 SPECTRE: a CEM Solver for Aircraft Design

We would like now to sketch some future work applying dual Schur complement methods to CEM problems. SPECTRE is a three-dimensional solver for the Maxwell equations developed at Dassault-Aviation for the last 6 years. We refer to [CLL+90] and [CZJ96] as well as the bibliography therein for a complete description of its ingredients, which we would like now to review briefly.

Governing Equations

Let $\boldsymbol{H}, \boldsymbol{E}, \boldsymbol{K}, \boldsymbol{J}$ denote respectively the magnetic field, the electric field, the magnetic surface current and the electric surface current. SPECTRE solves the time-harmonic Maxwell equations for perfectly conducting material which read, for domain Ω:

$$\begin{aligned} \nabla \times \nabla \times \boldsymbol{H} - k^2 \boldsymbol{H} &= 0 \quad \text{in} \quad \Omega, && (4.11) \\ \nabla \times \nabla \times \boldsymbol{E} - k^2 \boldsymbol{E} &= 0 \quad \text{in} \quad \Omega, && (4.12) \\ -\boldsymbol{n} \times \boldsymbol{E} &= \boldsymbol{K} \quad \text{on} \quad \partial\Omega, && (4.13) \\ \boldsymbol{n} \times \boldsymbol{H} &= \boldsymbol{J} \quad \text{on} \quad \partial\Omega, && (4.14) \end{aligned}$$

where $\boldsymbol{n}$ is the outward unit normal to $\partial\Omega$ and k is the incident wave number. In the far field, the Sommerfeld radiation condition should also be enforced.

Numerical Approximation

This radiation condition constitutes a major numerical difficulty in Maxwell equations. In the CEM community, it is customary to distinguish two types of problems:

interior problems: such as simulating fields in waveguides and cavities; there is no need for a radiation condition.

exterior problems: such as simulating fields scattered or radiated from structures; the Sommerfeld condition must be enforced.

For aircraft design, the above two situations are present. As proposed in [CZJ96], SPECTRE partitions domain Ω into two parts separated by a bounding surface and uses a coupling method between:

EFIE: an Electric Field Integral Equation solution, *exterior to the bounding surface*, that exactly enforces the Sommerfeld radiation condition. Piecewise linear electric surface currents $\boldsymbol{J}$ are defined on the boundary discretized by an unstructured surface mesh composed of triangles, leading to the following equation:

$$\boldsymbol{Z}_J \boldsymbol{J} = \boldsymbol{V}_i \qquad (4.15)$$

where $\boldsymbol{Z}_J$ is a dense matrix and $\boldsymbol{V}_i$ represents the incident field. A direct Gauss solver is used for the solution of (4.15).

MFVE: a Magnetic Field Volumic Equation solution, *interior to the bounding surface*. For a given electric field $\boldsymbol{E}$ computed from (4.15), equation (4.11) along with its natural boundary condition (4.13) is solved for magnetic field $\boldsymbol{H}$. A finite element formulation is used with an unstructured volumetric mesh composed of tetrahedra; it is based on the H(rot) tetrahedral element (see [CZJ96]) with unknowns defined on *edges*. The resulting linear system, with a sparse symmetric and complex matrix, is solved by a QMR iterative procedure with a SSOR preconditioner (see [FN91] and [BBC$^+$94]).

The above coupling is done in a *direct* manner. The bounding interface unknowns are eliminated by a succession of *independent* linear solves for MFVE, one for each degree of freedom on this interface.

MFVE: Parallel Implementation using Dual Schur Methods

For these solutions, we would like to apply the different parallel solvers defined in the previous section. In this perspective, we need to keep in mind the following differences with the CFD case:

1. The level of convergence required is higher: 10^{-8}.
2. Preliminary tests have shown that, due to the particular spectrum of the operator, the "outer" GCR algorithm in the dual Schur methods should be replaced by a QMR procedure.
3. We are solving for a given linear operator with a large number of independent right-hand sides: reconjugation techniques proposed in [Rou94] should be useful.

In light of these remarks, solver 0 seems to have the advantage.

5 Conclusion

We have proposed a parallel preconditioner, based on dual Schur methods and incomplete LDU factorization, which seems to be scalable both in terms of the number of subdomains as in terms of the level of convergence required.

Applications to problems in CFD have given thus far only a small gain compared to the usual diagonal preconditioner. We are expecting higher gains as we go on to "stiffer" problems such as those encountered in unsteady flows or multi-disciplinary optimization.

Applications to problems in CEM are still in the development stage.

REFERENCES

[BBC$^+$94] Barret R., Berry M., Chan T., Demmel J., Donato J., Dongarraa J., Eijkhout V., Pozo R., Romine C., and van der Vorst H. (1994) *Templates for the solution of linear systems : building blocks for iterative methods.* SIAM.

[CLL$^+$90] Calnibalosky C., Leflour G., Lohat P., Lombard J., Quadri J., and Vukadinovic N. (November 1990) Electromagnetic calculation of a whole aircraft by the code spectre. In *Proc. of JINA 90*, pages 83–87. CNET-IEEE, Nice.

[CMR94] Chalot F., Mallet M., and Ravachol M. (January 1994) A comprehensive finite element navier-stokes solver for low and high-speed aircraft design. *AIAA 94-814* .

[CZJ96] Cwik T., Zuffada C., and Jamnejad V. (April 1996) Modeling three-dimensional scatterers using a coupled finite element-integral equation formulation. *IEEE Trans. Antennas and Propagat.* 44(4): 453–459.

[FMR94] Farhat C., Mandel J., and Roux F. (1994) Optimal convergence properties of the finite element tearing and interconnecting domain decomposition method. *Comp. Meths. Appl. Mech. Eng.* (115): 365–385.

[FN91] Freund R. and Nachtigal N. (1991) A quasi-minimal residual method for non-hermitian linear systems. *Numerische Mathematik* 60: 315–339.

[Rou94] Roux F. (April 1994) Parallel implementation of a domain-decomposition method for non-linear elasticity problems. In D.Keyes Y.Saad D. (ed) *Proc.of the Workshop on Domain-based Parallelism and Problem Decomposition methods in Comp. Sci. and Eng.*, pages 161–175. SIAM, Minneapolis.

[Ven94] Venkatakrishnan V. (January 1994) Parallel implicit unstructured euler solvers. *AIAA J.* 32(10): 1985–1991.

84

Development of a Domain Decomposition Method for Computational Aeroacoustics

G. S. Djambazov, C.-H. Lai, and K. A. Pericleous

1 Introduction

Computational Aeroacoustics (CAA) implies the direct simulation of acoustic fields generated by flows and of the interaction of acoustic fields with flows. 'Direct' implies that the computation is only based on fundamental physical principles without reliance on empirical results or heuristic conjectures.

Since sound can be represented as comprised of different frequencies it is not difficult to estimate the resolution requirements for a typical domain, say around an airport, and a typical frequency range of interest, say up to about 100 Hz. If the ecologically important zone — immediately after take-off — is assumed to be 300 m long, 300 m wide, and 100 m high, and if 10 is assumed to be a reasonable number of grid points per wavelength, then the total number of grid points in this computational domain will be about 230 million. This shows that for realistic problems the computing power required with a direct approach is exceedingly large.

The classical approach to aerodynamic noise is Lighthill's "Acoustic Analogy" [Lig52] which does not involve direct computation of the sound field. It can be used to estimate the level of noise generated by some turbulent flows, but not the nonlinear interaction between the flow and the sound waves.

Since fluid flow can be easily computed using Computational Fluid Dynamics (CFD) techniques it is desirable to make use of existing CFD codes as much as possible. The region of interest is then most often split in two: 'near field' with nonlinear acoustic effects, and 'far field' of linear acoustics [SHM95, SH93]. CFD is meant to be used in the near field. In fact most CFD codes suffer from false (numerical) diffusion and will tend to lose the noise too near the place it is generated. Hence, they have to be combined with an Acoustic Analogy, which easily handles the far field [SH93, ZRL95].

If there is a code to compute the near field accurately enough, Kirchhoff's surface integral method can be applied to the far field [Lyr93]. Accordingly, any quantity which follows the wave equation outside a given surface is defined by its values, spatial and

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

temporal derivatives on the same surface. The Kirchhoff surface should be chosen far enough to contain all the nonlinearities of the near field, and still close enough to form a reasonably sized computational domain.

This paper describes the initial stages of the development of a CAA code for the near field. Following the domain decomposition approach certain modules of this code are shown to be useful in the far field as well.

2 Decomposition of the Variables

Sound is a form of fluid motion, and as such it is governed by the equations of continuity (2.1) and momentum (2.2). Any fluid quantity has been represented as a sum of a 'mean' value (indexed with $_0$), and a 'perturbation' value with no index. The index summation convention is in use:

$$\frac{\partial(\rho_0+\rho)}{\partial t}+\frac{\partial}{\partial x_j}\left[(\rho_0+\rho)(v_{j0}+v_j)\right]=0, \tag{2.1}$$

$$(\rho_0+\rho)\left[\frac{\partial(v_{i0}+v_i)}{\partial t}+(v_{j0}+v_j)\frac{\partial(v_{i0}+v_i)}{\partial x_j}\right]+\frac{\partial(p_0+p)}{\partial x_i}=F_{i0}+F_i. \tag{2.2}$$

The symbols ρ, v, p denote respectively the fluid density, velocity and pressure while F contains both external forces and internal friction.

There are good reasons for splitting the variables [VS95]. Physically, sound waves are small perturbations to other large scale motions, such as wind, and numerically they are best represented separately. Also, in this way existing CFD codes can be used to solve the 'mean flow'.

After expanding the brackets many new terms appear in the governing equations, and an analysis has been made for the magnitude of each of these. For a typical aeroacoustic problem, terms containing a perturbation quantity are very small compared to the rest; terms with derivatives of perturbation quantities are not necessarily so. When all the small terms are moved to the right-hand side, equations (2.3) and (2.4) are obtained:

$$\frac{\partial\rho}{\partial t}+v_{j0}\frac{\partial\rho}{\partial x_j}+\rho_0\frac{\partial v_j}{\partial x_j}=\cdots=q, \tag{2.3}$$

$$\rho_0\left(\frac{\partial v_i}{\partial t}+v_{j0}\frac{\partial v_i}{\partial x_j}\right)+\frac{\partial p}{\partial x_i}=F_i-\cdots=f_i. \tag{2.4}$$

These will be the governing equations for the CAA algorithm. With the mean-flow values defined by CFD the left-hand side is linear, and the small right-hand side can be solved for iteratively. Based on the above analysis fast convergence is expected.

The interaction between CFD and CAA is pictured in Figure 1. It is generally assumed that back-reaction of the sound on the flow field is only to be expected when there is a resonator nearby [Lig52], and that is denoted by the dotted arrow. In most cases iterations in that outermost loop will not be necessary.

Figure 1 Interaction between CFD and CAA codes

Mean flow:

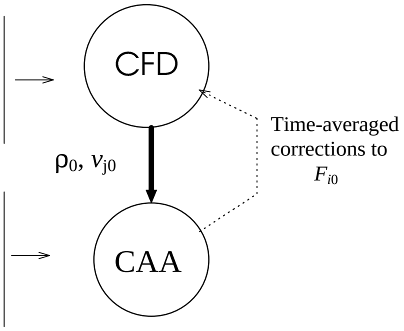
$$\frac{\partial \rho_o}{\partial t} + \frac{\partial}{\partial x_j}(\rho_o v_o) = 0$$

$$\rho_o\left(\frac{\partial v_{io}}{\partial t} + v_{jo}\frac{\partial v_{io}}{\partial x_j}\right) + \frac{\partial p_o}{\partial x_i} = F_{io}$$

Acoustic perturbation:

$$\frac{\partial \rho}{\partial t} + v_{jo}\frac{\partial \rho}{\partial x_j} + \rho_o\frac{\partial v_j}{\partial x_j} = q$$

$$\rho_o\left(\frac{\partial v_i}{\partial t} + v_{jo}\frac{\partial v_i}{\partial x_j}\right) + \frac{\partial p}{\partial x_i} = f_i$$



3 Linear Propagation Solver

The discussion above has made it clear that to study the generation of aerodynamic noise one has to resolve also its propagation, which occurs simultaneously. That can be done by a hyperbolic solver, which will become a module in a nonlinear code for the sound-generation zone. It can also be used on its own for the propagation zone, where the linearized Euler equations [SHM95] are most often assumed.

Using the standard definition for the speed of sound, c, the framework for a two-dimensional simulation is presented below:

$$\begin{aligned}
\frac{\partial p}{\partial t} &+ c\left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}\right) = c^2 q, && (3.5)\\
\frac{\partial u}{\partial t} &+ c\frac{\partial p}{\partial x} = c f_x\,,\ u = \rho_0 c v_x, && (3.6)\\
\frac{\partial v}{\partial t} &+ c\frac{\partial p}{\partial y} = c f_y\,,\ v = \rho_0 c v_y, && (3.7)\\
\frac{\partial p}{\partial \rho} &= c^2 = \gamma\frac{p_0}{\rho_0}\ (\gamma_{air} = 1.4). &&
\end{aligned}$$

Perturbation velocity components along the axes x and y are denoted v_x and v_y, respectively. The right-hand sides are now considered known. At any time step a relative frame of reference — moving with the mean flow — is considered, hence there are no convection terms left.

The finite volume approach has been followed for the discretization of these equations. Successive integration along x, y, and t using the notation of Figure 2

Figure 2 Finite volume formulation – cell-face symbols

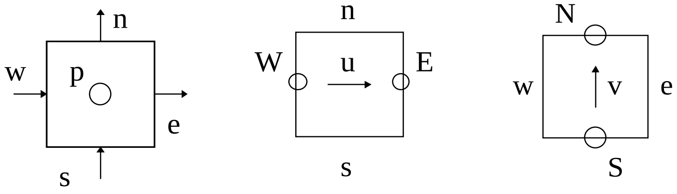
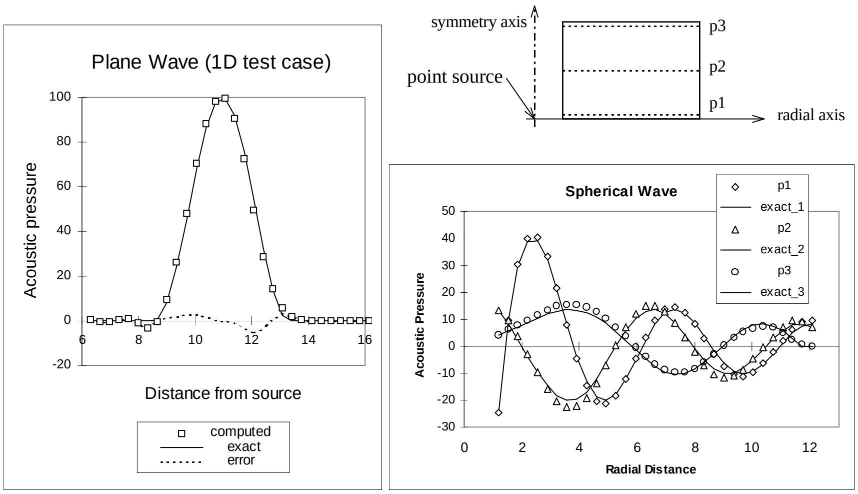


Figure 3 Sample results for the fully explicit numerical scheme

56 time steps, Courant number = 0.7



yields:

$$\int\limits_{cell} (p - p_{old})dxdy + c\Delta t \left[\int\limits_{s}^{n} (u_e - u_w)dy + \int\limits_{w}^{e} (v_n - v_s)dx \right] = c^2 \Delta t \int\limits_{cell} q\, dxdy,$$

$$\int\limits_{cell} (u - u_{old})dxdy + c\Delta t \int\limits_{s}^{n} (p_E - p_W)dy = c\Delta t \int\limits_{cell} f_x dxdy,$$

$$\int\limits_{cell} (v - v_{old})dxdy + c\Delta t \int\limits_{w}^{e} (p_N - p_S)dx = c\Delta t \int\limits_{cell} f_y dxdy.$$

Integration along the time axis is expressed here through mean values for simplicity.

The Fully Explicit Numerical Scheme

Scalar quantities (i.e., pressure) are stored in the centres of the finite-volume cells while vector quantities (i.e., velocity components) are stored at the cell faces in the middle of the time steps. All integrals are approximated by stepwise functions. The resulting explicit scheme is then the same as the one resulting from a finite-difference approach:

$$\begin{aligned} p = p_{old} - Cou[(u_e - u_w) + (v_n - v_s)] + c^2\Delta t q, \\ u = u_{old} - Cou(p_E - p_W) + c\Delta t f_x, \\ v = v_{old} - Cou(p_N - p_S) + c\Delta t f_y, \\ Cou = c\Delta t/h, \end{aligned}$$

where Δt is the length of the time step, and h is the size of the cells on a regular mesh.

This scheme was first tested in one dimension, which is the case of plane wave propagation (with $v = 0$) because exact analytical solutions can be obtained for plane waves (1D) and spherical waves (3D), but not for circular waves (2D). Computed results were exact at the limit of stability with Courant number $Cou = 1$. As these are of no practical interest, further tests were done with the maximum Courant number for the 2-dimensional case, i.e., 0.7 (see left graph of Figure 3).

An axisymmetric version of the test code was developed to check the resolution of spherical waves, and the results are pictured to the right-hand side of the same figure. The 56 time steps in these tests correspond to 39 cell-lengths and about 3 wavelengths.

The Improved Numerical Scheme

Although the graphs in Figure 3 seem encouraging the accuracy is actually not sufficient as the error increases with each time step, and after 200 - 300 time steps it becomes unacceptable.

If there are no shocks and the sound field is described by continuous functions, the best way to overcome this difficulty is higher order approximation. Bearing in mind that the final code will be 3-dimensional involving too many neighbours in the numerical scheme does not seem convenient. Based on these arguments a parabolic approximation has been applied to all functions in the finite-volume integrals:

$$\int_{x_0-\frac{h}{2}}^{x_0+\frac{h}{2}} f(x)dx = h[Af(x_0 - h) + (1 - 2A)f(x_0) + Af(x_0 + h)]\,,\; A = \frac{1}{24}.$$

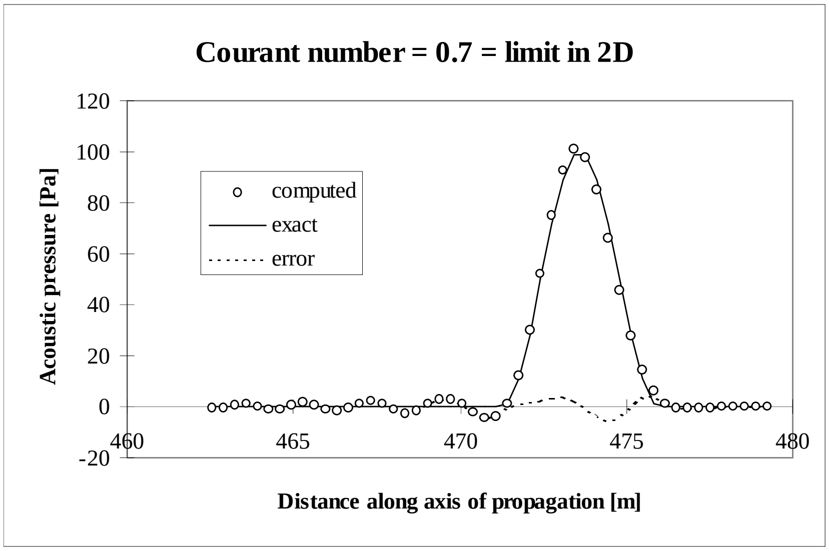
In 2D this means

$$\int_{cell} f(x,y)dxdy = A\sum_{nb} f_{nb} + (1 - 4A)f_{cell},$$

with 'nb' used to denote all four neighbours of a cell in a regular mesh.

The numerical scheme now becomes implicit, so a linear system of equations has to be solved at each time step. Direct methods cannot be considered because of the

Figure 4 Validation of the improved numerical scheme in 1D for *2000* time steps



large number of grid points. However, the iterative solution can be provided with a very good initial guess based on the explicit scheme.

When finally the nonlinear terms are implemented in the code their iterations can be done simultaneously with the solution of the linear system.

Tests were performed on the new version in 1D over 2000 time steps. These are equivalent to about 100 wavelengths and seem to be enough to take the signal studied out of the near field. The results shown in Figure 4 exhibit encouraging accuracy. The amplitude of the oscillating numerical error will be about 5% at the boundaries of the near field, which is acceptable. As expected, the efficiency of the algorithm is very high: only 3 to 4 iterations are necessary for convergence at each time step.

4 Coupling of the Subdomains

Since the noise generation domain (near field) is always contained in the propagation domain (far field) all that has to be done is to implement adequate radiating boundary conditions for the near field and to solve it prior to the far field. Time-dependent values of the variables are then imposed on the inner boundaries of the far field, and radiation conditions on its outer boundaries.

With the test version of the code the boundaries have been chosen to be comprised of cell faces (stepwise boundaries). Thus only the time-dependent velocity component that is perpendicular to the boundary has to be determined. This has been done under the assumption that the wave equation holds there (no sources of sound at the boundary), and that the outgoing waves are plane waves. The boundary velocities then can be determined from the velocity field at the previous time step by interpolation. The exact positions of the interpolation points depend on the directions of acoustic radiation and on the Courant number, and can be calculated for each boundary face.

5 Conclusions and Further Work

A combined decomposition method of both the domain and the variables has been developed that is capable of handling the computationally intensive problems of aeroacoustics.

A solver for the linearized Euler equations has been implemented and tested showing encouraging accuracy. It is a module to be used in both domains for the perturbation variables. Based on a finite-volume staggered grid, the algorithm proposed is stable and efficient.

Future work includes three main stages. First: implementation of the convection terms from (2.3) and (2.4), which have been omitted in this study. (These results will be presented at the 3rd AIAA/CEAS Aeroacoustics Conference in May 1997.) Second: discretization and implementation of the nonlinear terms to complete the acoustic module for the near field. Third: coupling of the acoustic module with a CFD code. In order to build an efficient coupled algorithm, a study has to be done to determine how fine the CFD mesh and time stepping has to be, and whether to include the viscous terms in the acoustic solver.

The combined CFD-CAA code will find environmental applications in studying the mechanisms of generation of aerodynamic noise, and in searching for a way of reducing the noise made by jets, propellers, and wind.

REFERENCES

[Lig52] Lighthill J. M. (1952) On sound generated aerodynamically: I. general theory. In *Proceedings of The Royal Society*, volume 211 A, pages 564–587. London.

[Lyr93] Lyrintzis A. S. (June 1993) The use of kirchhoff's method in computational aeroacoustics. In R.R.Mankbadi, A.S.Lyrintzis, O.Baysal, L.A.Povinelli, and M.Y.Hussaini (eds) *Computational Aero- and Hydro-acoustics*, number 147 in FED, pages 53–61. ASME, New York.

[SH93] Sarkar S. and Hussaini M. (June 1993) A hybrid direct numerical simulation of sound radiated from isotropic turbulence. In R.R.Mankbadi, A.S.Lyrintzis, O.Baysal, L.A.Povinelli, and M.Y.Hussaini (eds) *Computational Aero- and Hydro-acoustics*, number 147 in FED, pages 83–89. ASME, New York.

[SHM95] Shih S., Hixon D., and Mankbadi R. (1995) A zonal approach for prediction of jet noise. *CEAS/AIAA Paper 95-144* .

[VS95] Viswanathan K. and Sankar L. (1995) Numerical simulation of airfoil noise. In A.S.Lyrintzis, R.R.Mankbadi, O.Baysal, and M.Ikegawa (eds) *Computational Aeroacoustics*, number 219 in FED, pages 65–70. ASME, New York.

[ZRL95] Zhang X., Rona A., and Lilley G. (1995) Far-field noise radiation from an unsteady supersonic cavity flow. *CEAS/AIAA Paper 95-040* .

85

On Domain Decomposition for a Three-dimensional Extrusion Model

M. S. Eikemo

1 Introduction

The thermo-mechanical properties of aluminium during an extrusion process is described by a coupled set of nonlinear partial differential equations. The model is three-dimensional in order to support practical applications, and consists of a temperature equation, a continuity equation, and Navier-Stokes equations with a nonlinear Zener-Hollomon material law. The convective part is discretized in a Lagrangian sense using a modified method of characteristics. The equations are linearized in a straightforward manner, and we use a mixed finite element discretization with quadratic hexahedral elements for the approximation of velocities and linear hexahedral elements for temperature and pressure. After some decoupling, a positive definite system of linear equations for temperature and an indefinite block system for velocity and pressure are obtained. We use overlapping Schwarz domain decomposition methods in combination with a Krylov subspace accelerator to solve the problem. These techniques are powerful methods for solving problems on complex geometries, as they allow the possibility of local refinement at locations where the systems experience large gradients.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 Model Description

For a domain $\Omega \subset R^3$ the fully scaled system of equations describing an extrusion process is given as

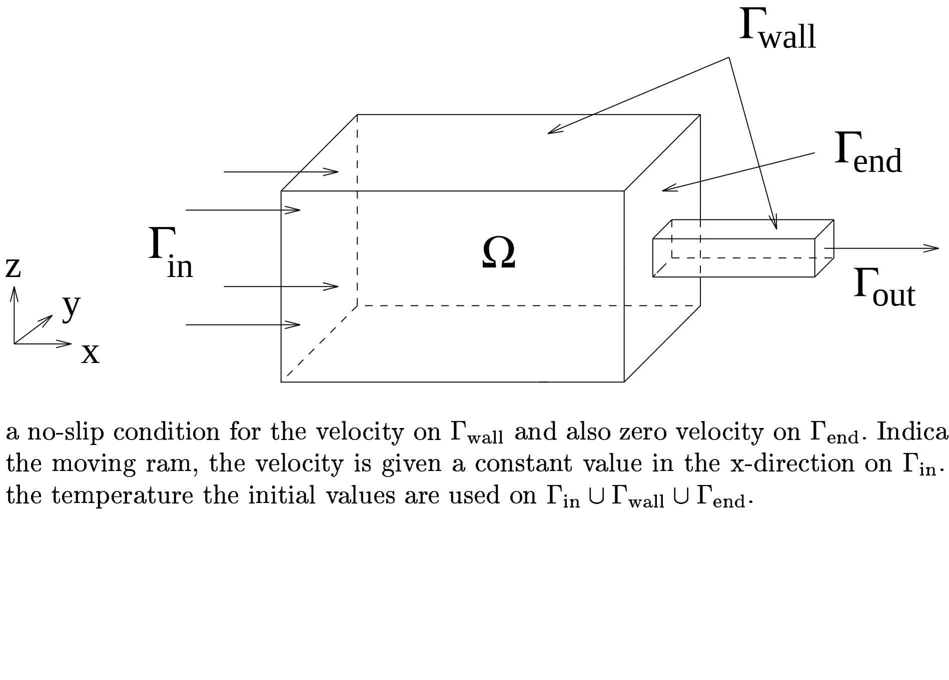
$$\begin{aligned} \mathrm{Re}(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}) &= -\nabla p + \nabla\cdot\tau && \text{in } \Omega, \\ \nabla\cdot\mathbf{u} &= 0 && \text{in } \Omega, \\ \mathrm{Pe}(\frac{\partial \theta}{\partial t} + \mathbf{u}\cdot\nabla\theta) &= \nabla^2\theta + \beta\epsilon : \tau && \text{in } \Omega, \end{aligned} \tag{2.1}$$

where dyadic notation is used for the last term, $\epsilon : \tau = \epsilon_{ij}\tau_{ij}$. The primary variables are the velocity $\mathbf{u} = (u_1\ u_2\ u_3)^T$, the pressure p, and the temperature θ. Further, $\tau = 2\mu\epsilon$ is the stress tensor, where $\mu = \bar{\tau}/(3\bar{\epsilon})$ is the nonlinear viscosity coefficient. The effective strain rate is given by $\bar{\epsilon} = (\frac{2}{3}\epsilon : \epsilon)^{\frac{1}{2}}$, where $\epsilon = \frac{1}{2}(\nabla\mathbf{u} + \nabla\mathbf{u}^T)$ is the strain rate tensor. The viscosity coefficient also contains the Zener-Hollomon material model $\bar{\tau}(\bar{\epsilon},\theta) = \alpha^{-1}\mathrm{arcsinh}((Z/K)^{\frac{1}{m}})$, where α, A and m are material parameters and Z is the Zener-Hollomon parameter given by $Z = Z(\bar{\epsilon},\theta) = \bar{\epsilon}\exp(Q/(R\theta))$, with Q denoting activation energy and R denoting the universal gas constant. The Reynolds number and the Péclét number are denoted by Re and Pe, respectively. For a more thorough presentation of the equations and the scaling procedure we refer to [Eik96], and topics concerning the extrusion process itself are discussed in [HSH92, HGS92] and references therein.

From specific material- and problem-dependent parameters we get the Reynolds number to be very small, typical of magnitude 10^{-8}. We may therefore neglect the left side of the vector equation in (2.1). Note that the equation still is nonlinear through the stress tensor.

Our computational domain is shown in Figure 1 and has a narrow channel to mimic the effect of an extrusion tool and a boundary $\partial\Omega = \Gamma_{\text{in}} \cup \Gamma_{\text{out}} \cup \Gamma_{\text{wall}} \cup \Gamma_{\text{end}}$. We use a no-slip condition for the velocity on Γ_{wall} and also zero velocity on Γ_{end}. Indicating the moving ram, the velocity is given a constant value in the x-direction on Γ_{in}. For the temperature the initial values are used on $\Gamma_{\text{in}} \cup \Gamma_{\text{wall}} \cup \Gamma_{\text{end}}$.

Figure 1 Computational domain, with $\Gamma_{\text{wall}} = \partial\Omega \backslash (\Gamma_{\text{in}} \cup \Gamma_{\text{out}} \cup \Gamma_{\text{end}})$.



The pressure and velocity are slowly varying over a time step relative the variation of the temperature. We decouple the heat balance equation from the other equations by, on each time level, using a sequential iterative solution procedure. The boundary value problem for velocity-pressure can be stated as follows,

$$\begin{array}{rcll} -\nabla\cdot\tau+\nabla p &=& 0 & \text{in } \Omega, \\ \nabla\cdot\mathbf{u} &=& 0 & \text{in } \Omega, \\ \mathbf{u}(\mathbf{x}) &=& \mathbf{f}(\mathbf{x}) & \text{on } \partial\Omega\backslash\Gamma_{\text{out}}, \\ p(\mathbf{x}) &=& 0 & \text{on } \Gamma_{\text{out}}. \end{array} \tag{2.2}$$

and the initial boundary value problem for the temperature, with I being a time interval,

$$\begin{array}{rcll} \text{Pe}(\dfrac{\partial\theta}{\partial t}+\mathbf{u}\cdot\nabla\theta) &=& \nabla^2\theta+\beta\epsilon:\tau & \text{in } I\times\Omega, \\ \mathbf{u}(\mathbf{x}) &=& \mathbf{f}(\mathbf{x}) & \text{in } \Omega, \\ \theta(\mathbf{x},t) &=& g(\mathbf{x}) & \text{on } \partial\Omega\backslash\Gamma_{\text{out}}\ \forall t, \\ \theta(\mathbf{x},0) &=& \theta_0(\mathbf{x}) & \text{in } \Omega \text{ at } t=0. \end{array} \tag{2.3}$$

3 Solution Procedure

The discretization for the velocity-pressure system (2.2) is carried out by a mixed finite element method, [Cia78, DJEW83, GR86, ELRV89, DES92], using hexahedral elements with triquadratic approximation for each of the velocity components and trilinear approximation for pressure, also called the hexahedral version of the Q_2-Q_1 element. The linearization is handled by Picard iterations.

Carrying out the procedure described in [Eik96] results in a linear system $M\mathbf{y}=\mathbf{b}$, where the stiffness matrix has a block structure,

$$\left\{\begin{array}{cccc} A_{11} & A_{12} & A_{13} & B_1^T \\ A_{21} & A_{22} & A_{23} & B_2^T \\ A_{31} & A_{32} & A_{33} & B_3^T \\ B_1 & B_2 & B_3 & 0 \end{array}\right\}\left\{\begin{array}{c} X \\ Y \end{array}\right\}=\left\{\begin{array}{cc} A & B^T \\ B & 0 \end{array}\right\}\left\{\begin{array}{c} X \\ Y \end{array}\right\}=\left\{\begin{array}{c} F \\ G \end{array}\right\}. \tag{3.4}$$

The matrix $A=\{A_{ij}\}$, $i,j=1,2,3$, is a 3-by-3 block matrix corresponding to the velocity components in the three momentum equations in (2.2). Since the matrix M is indefinite, the system (3.4) has to be reformulated in order for the preconditioned conjugate gradient method (PCG) to be applicable. In [BP94] a block preconditioning technique are introduced and we show in [Eik96] that this technique is efficient also for the extrusion problem. After some algebra on the rows, the system (3.4) is reformulated in such a way that the new coefficient matrix $\tilde{M}$ is positive definite,

$$\tilde{M}\left\{\begin{array}{c} X \\ Y \end{array}\right\}=\left\{\begin{array}{cc} A_0^{-1}A & A_0^{-1}B^T \\ BA_0^{-1}(A-A_0) & BA_0^{-1}B^T \end{array}\right\}\left\{\begin{array}{c} X \\ Y \end{array}\right\}=\left\{\begin{array}{c} A_0^{-1}F \\ BA_0^{-1}F-G \end{array}\right\}, \tag{3.5}$$

where A_0 is a preconditioner for A. The PCG method is now applicable, and as a preconditioner for this system,

$$\tilde{M}_0 = \left\{ \begin{array}{cc} I & 0 \\ 0 & \mathcal{K} \end{array} \right\} \tag{3.6}$$

is used, where I is the identity matrix and $\mathcal{K}$ is the preconditioner for the Schur complement $BA^{-1}B^T$,

$$\mathcal{K} = N_h + h^2 I, \tag{3.7}$$

where h is the spatial resolution and N_h is the solution operator on the pressure grid for a finite element approximation to a Neumann problem, that is $w = N_h f$ satisfies $(\nabla w, \nabla \phi) = (f, \phi)$ for test functions ϕ. The theory in [BP94] shows that this preconditioner gives rise to convergence rates which can be bounded independently of the mesh size h. As a preconditioner for A we use a block-diagonal matrix A_0^{-1} with three copies of a preconditioner for the submatrix A_{11} on the diagonal, see [Eik96]. We have mostly been using incomplete factorization as preconditioner for A_{11}.

The discretization procedure for the temperature problem (2.3) is carried out in two steps. First, the hyperbolic part is solved by the modified method of characteristics (MMOC) and second, the resulting elliptic problem is discretized by the finite element method. Following the MMOC scheme in [Eik96] we get an elliptic equation with a known right-hand side, and the finite element approach results in a linear system

$$A\mathbf{x} = \mathbf{b}, \tag{3.8}$$

where the matrix A is positive definite and the PCG method is applicable. We have used both incomplete factorization and multigrid cycles as preconditioners.

4 Domain Decomposition

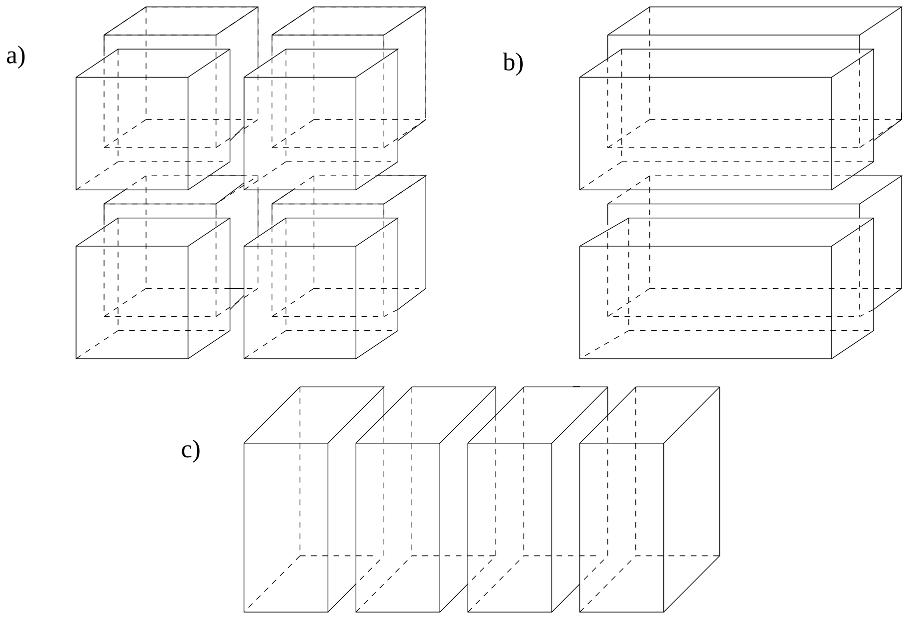
The extrusion problem is rich on localized phenomena, and domain decomposition methods prepare for local grid refinement. Near the extrusion tool at the outlet, with its complex structure with bridges and channels, large gradients in the flow pattern is produced. In order to capture these effects, the mesh is refined in these specific regions. In this way the size of the problem can be held at a minimum even with a high resolution in critical regions. Below we first show results from using additive and multiplicative Schwarz for different decompositions, and then present a local grid refinement technique. We refer to [BGS96] and references therein for a presentation of Schwarz methods and to [Sæv90, Sæv93, BEPS88, DEES90] for local grid refinement techniques.

Let the finite element space V^h be represented by the sum of N subspaces,

$$V^h = V_1^h + ... + V_N^h, \tag{4.9}$$

where N is the number of subdomains, Ω_i. Following the presentation of the methods in [Eik96], the multiplicative and additive Schwarz methods take the forms of iterative

Figure 2 Different decompositions, a) (2,2,2), b) (1,2,2) and c) (4,1,1).



methods for solving

$$(I - \prod_{i=1}^{N}(I - P_i))u_h = g_m \quad \text{and} \quad (\sum_{i=1}^{N} P_i)u_h = g_a, \tag{4.10}$$

respectively. Here $P_i \ : \ V^h \rightarrow V_i^h$ are orthogonal projections with respect to the bilinear form $a(.,.)$, and g_m and g_a are appropriate right-hand sides. The bilinear form $a(.,.)$ is appearing in the variational formulation of the problem to be solved, i.e. $a(u_h, \phi) = F(\phi)$ for test functions ϕ. For the heat equation it is defined like

$$a(\theta_h, \phi) = (\theta_h^{n+1,m+1}, \phi) + \frac{\Delta t}{\mathrm{Pe}}(\nabla\theta_h^{n+1,m+1}, \nabla\phi) - \frac{\Delta t}{\mathrm{Pe}}((\beta\epsilon : \tau)_h^{n+1,m}, \phi) \tag{4.11}$$

where $(.,.)$ denotes the usual L_2-inner product and the superscript m counts the Picard iterations and n indicates the time step. Figure 2 shows examples of different ways to divide the domain Ω into subdomains Ω_i. The decomposition indicated by the pair (k, l, m) means to divide the domain into k parts along the x-direction, l parts along the y-direction and m parts along the z-direction, giving a total of $k \times l \times m$ subdomains. Consider the temperature problem (2.3). Table 1 gives results in terms of number of preconditioned conjugate gradient (PCG) iterations and condition number, κ, for both additive (AS) and symmetric multiplicative Schwarz (SMS) used as preconditioner. The time step is $\Delta t = 0.1$. The condition number is calculated from the Ritz values, see [Eik96], and the iterations are terminated when the discrete L_2-norm of the residual is reduced by a factor $\epsilon_{CG} = 5 \cdot 10^{-8}$. The experiments support

Table 1 Number of PCG iterations with symmetric multiplicative Schwarz (SMS) and additive Schwarz (AS) preconditioners, and corresponding condition numbers, κ_m and κ_a. The three first columns give the number of elements in the x-, y- and z-directions, the kind of decomposition according to Figure 2 and the number of overlapping elements, respectively.

elements	decomposition	overlap	SMS	κ_m	AS	κ_a
32×32×32	(2,2,2)	1	5	1.41	14	22.41
		2	3	1.08	12	13.18
		3	3	1.02	10	10.58
	(1,2,2)	1	5	1.41	15	22.56
		2	3	1.07	12	13.24
		3	3	1.02	10	10.59
	(1,1,2)	1	5	1.34	10	6.24
		2	3	1.06	7	4.11
		3	2	1.01	6	3.49
	(2,1,1)	1	4	1.32	6	2.93
		2	2	1.05	4	2.58
		3	2	1.01	4	2.26
	(4,1,1)	1	4	1.37	7	3.11
		2	3	1.07	5	2.71
		3	2	1.02	5	2.31

the well known properties of the methods, see [Eik96]. Note also that the table show that the (1,2,2) decomposition has more in common with the (2,2,2) than with the (4,1,1), which is the other four-subdomain decomposition. This is reasonable since all the subdomains overlap each other in both cases. We also observe from the table that the results get worse when the decomposition has a division in the z-direction, especially for the additive preconditioner. By calculating the temperature field on Γ_{out} we invoke a Neumann condition on this boundary, and this condition, compared to a Dirichlet condition, makes greater demands on the system. Decompositions in the y- and z-directions result in several subdomains with a Neumann boundary, while decomposition in the x-direction only causes only one subdomain to have a Neumann boundary. This will be a topic for further investigation.

Consider now $\Omega \subset R^3$ to be the domain in Figure 3, consisting of a body and a narrow channel. We begin by introducing a coarse grid for Ω, with mesh size h_c. Then a fine grid according to a refinement level k is introduced, for which the mesh size is $h_f = 2^{-k}h_c$. The domain is divided into two subdomains, one covered with the coarse mesh and one with the fine mesh, see Figure 3. The triangulation leads to the introduction of a set of so-called slave nodes on the boundary surface of the refined region. The values of functions in the composite finite element space in these nodes are, because of the continuity assumption, completely determined by their values in neighbouring coarse-grid nodes. This also means that a discrete function is uniquely represented by a vector with entries corresponding to the genuine degrees of freedom,

Figure 3 Composite mesh for the domain $\Omega = \Omega_1 \cup \Omega_2$, $\partial\Omega = \Gamma_{\text{in}} \cup \Gamma_{\text{out}} \cup \Gamma_{\text{wall}} \cup \Gamma_{\text{end}}$.

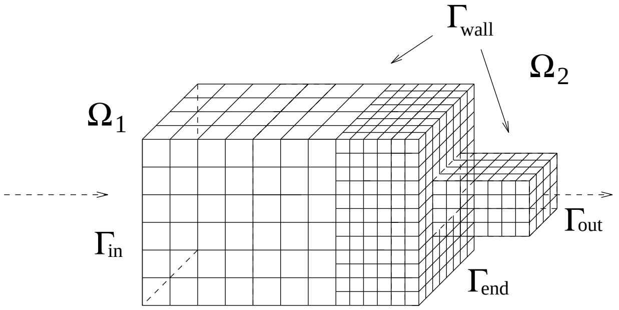
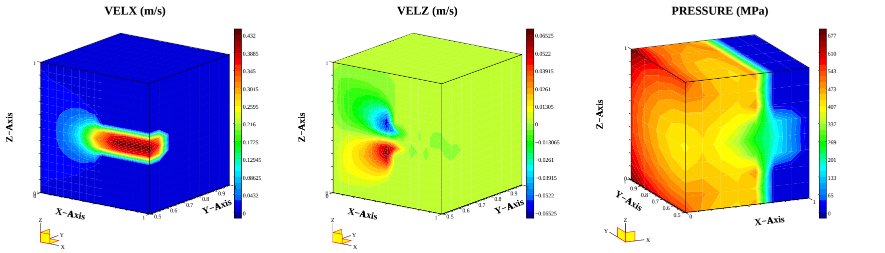


Figure 4 Domain consisting of 4 × 8 × 8 elements and a narrow channel of 3 × 2 × 2 elements. No-slip boundary condition **u**=0 on Γ_{wall}, $u_1 = 1$ (0.015 m/s) on Γ_{in}, $u_3 = 0$ on Γ_{in} and $p = 0$ on Γ_{out}. The three subfigures show the x-component of the velocity, the z-component of the velocity and the pressure, respectively. Note that the last domain is turned compared to the other two.



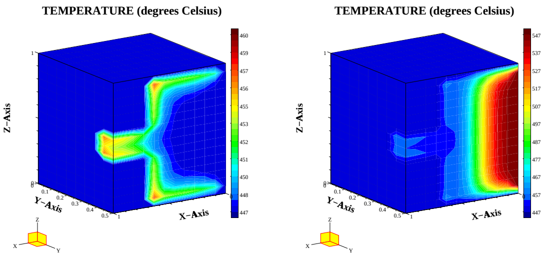
i.e. all nodes except the slave nodes. Let cross nodes denote the nodes on the interface which appear both in the coarse and the refined grid. By employing this refinement strategy the size of the problem is dramatically reduced compared to having a global fine grid.

We use the PCG method with SMS as a preconditioner. The subdomain structure can be utilized both in the matrix multiplication operation in the PCG algorithm and the preconditioning procedure. To avoid having to deal with the irregular global stiffness matrix, A, arising from the composite grid considered, the matrix is never explicitly completed. Instead it is considered as a set of submatrices, each submatrix referring to a certain subdomain Ω_i, see also [Sæv90]. Matrix multiplication will consist of operations on each of the subdomains separately, and then gluing the global product together along the interface. The SMS preconditioner utilizes in our case the same subdomains as was used to define the composite mesh, extending the refined region to define the overlap. In cases with more than one refined subdomain, however, the preconditioner may use subdomains consisting of some collection of refined regions. These procedures are given a more thorough presentation in [Eik96].

Results obtained with local refinement in the outlet region compared to global fine resolution are reported in [Eik96]. The dramatic changes in velocity occur in the outlet region and are properly resolved by the fine grid. For the temperature, where local behaviour is seen near Γ_{wall} in the whole domain, these effects are not properly resolved on a coarse grid. Similar for the pressure, rapid changes in values occur near the in-boundary. It is obvious that making the refinement technique adaptive will cause the effects to be properly resolved wherever they occur.

Figure 4 visualizes the solutions from solving (2.2) with $\mathbf{u} = (1\ 0\ 0)^T$ on Γ_{in} and $p = 0$ on Γ_{out}. The temperature is held constant at zero, i.e. 447 °C. Note that only half the domain is plotted in order to be able to see the behaviour of the solutions inside, along the flow direction. We see that u_1 has attained a domal profile with the largest values in the middle and decreasing to zero towards the surfaces making Γ_{wall}. This profile is also supported by the plot for u_3, where there are negative values in the upper half of the domain and positive values in the lower half, showing flow downwards from the top and up from the bottom. We observe the largest pressure towards the boundaries around the inlet area, and especially in the corners, due to the boundary conditions $u_1 = 1$ on Γ_{in} and $u_1 = 0$ on Γ_{wall}. Figure 5 shows temperature results from solving (2.3) with a given velocity field $u_1 = \sin(\pi y)\sin(\pi z)$ for two cases of boundary values, see the figure text. We observe the two main effects, transportation and very local changes in gradients.

Figure 5 Temperature solutions after two time steps, $\Delta t = 0.1$. Domain with body consisting of $8 \times 16 \times 16$ elements and channel of $3 \times 4 \times 4$ elements. Boundary condition $\theta = 0$ (447 °C) on Γ_{in} and $\theta = 0$ (547 °C) on Γ_{in}, respectively.



5 Conclusions and Future Work

In this paper we have reported results obtained from applying domain decomposition methods to the extrusion problem. Together with local mesh refinement these methods proved to be very efficient techniques for reducing the size of the problem and at the same time obtaining the required accuracy.

Future activity involves both improvements of the solution methods and extensions of the model. The most obvious and most necessary improvement of the solution strategy is to make the grid refinement adaptive. Our numerical experiments show the need for following temperature fronts in addition to handling more stationary local effects. Extensions of the model may be to involve more complex geometries and include input from the surroundings in terms of boundary conditions. A realistic die has a very complex geometry with hollow spaces and bridges and heat transfer between the metal and the container, ram and die do occur.

The programming language C++ offers several tools for supporting data abstraction and object-orientation. A further utilization of all the possibilities and subtleties of C++ would increase the efficiency of the implementation and make it applicable to more general models.

Acknowledgement

The support provided by Norsk Hydro is gratefully acknowledged.

REFERENCES

[BEPS88] Bramble J., Ewing R., Pasciak J., and Schatz A. (1988) *A preconditioning technique for the efficient solution of problems with local grid refinement.* Comp. Meth. Appl. Mech. Eng. 67: pp. 149–159.

[BGS96] Bjørstad P., Gropp W., and Smith B. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press.

[BP94] Bramble J. and Pasciak J. (1994) *Iterative techniques for time dependent Stokes problems.* Math. Comp. .

[Cia78] Ciarlet P. (1978) *The finite element method for elliptic problems.* North Holland Publishing Company.

[DEES90] Dahle H., Espedal M., Ewing R., and Sævareid O. (1990) *Characteristic Adaptive Subdomain Methods for Reservoir Flow Problems.* Numer. Meth. for Partial Differential Equations 6: pp. 279–309.

[DES92] Dahle H., Espedal M., and Sævareid O. (1992) *Characteristic local grid refinement for reservoir flow problems.* Int. J. Numer. Meth. Eng. 34: pp. 1051–1069.

[DJEW83] Douglas J., JR, Ewing R., and Wheeler M. (1983) *The approximation of the pressure by a mixed method in the simulation of miscible displacement. RAIRO Anal. Numer.* 17: pp. 17–33.

[Eik96] Eikemo M. (1996) *On Numerical Techniques and Characteristic Local Refinement for Simulating Aluminium Extrusion.* PhD thesis, Department of Mathematics, University of Bergen.

[ELRV89] Ewing R., Lazarov R., Russel T., and Vassilevski P. (1989) *Local Refinement via Domain Decomposition Techniques for Mixed Finite Element Methods with Rectangular Raviart-Thomas Elements.* In 3rd International Symposium on Domain Decomposition Methods for Partial Differential Equations. SIAM.

[GR86] Girault V. and Raviart P. (1986) *Finite Element Methods for Navier-Stokes Equations.* Springer-Verlag, Berlin Heidelberg.

[HGS92] Herberg J., Gundesø K., and Skauvik I. (1992) *Application of Numerical Simulations in Design of Extrusion Dies.* In 5th International Alu. Extr. Techn. Seminar. Chicago, Illinois, USA.

[HSH92] Holthe K., Støren S., and Hansen L. (1992) *Numerical simulation of the aluminium extrusion process in a series of press cycles.* In 4th International Conference on Numerical Methods in Industrial Forming Processes. Balkema.

[Sæv90] Sævareid O. (1990) *On Local Grid Refinement Techniques for Reservoir Flow Problems.* PhD thesis, Department of Mathematics, University of Bergen.

[Sæv93] Sævareid O. (1993) *Simulation of Extrusion combining Local Grid Refinement and Lagrangian time-marching.* In 8th International Conference on Finite Elements in Fluids. Pineridge Press, Barcelona.

Domain Decomposition Methods Applied to Sedimentary Basin Modeling

F. Willien, I. Faille, and F. Schneider

1 Introduction

Basin modeling aims to reconstruct the evolution of a sedimentary basin taking into account compaction of the porous medium, hydrocarbon formation, and migration. A sedimentary basin is a heterogeneous porous medium consisting of stacked stratigraphic layers that have been deposited from the start of the basin up to the present. Over time, solid organic material contained in some stratigraphic layers is transformed into mobile hydrocarbons under the effect of increased temperature. Then, hydrocarbons flow in the porous medium to accumulate in reservoirs. This migration takes place as a two-phase or three-phase flow.

Up to now, most of basin simulators have been able to handle relatively simple geometries resulting from deposition, erosion, and compaction of the porous medium [UBD$^+$90]. However, most of real life basins are cut by faults along which block displacements can occur. The aim of Ceres project[1] is to model three-phase flow in a 2D section of a basin, whose geometry changes due to deposition, compaction, erosion of sediments, and block displacements along faults. In order to handle these complex geometries, Domain Decomposition (DD for short) techniques have been chosen. Indeed, faults cut the basin into domains that naturally define computational subdomains. In this first stage of the project, a simplified problem is considered in which only one-phase flow can occur and where faults are vertical. Equations that govern the physical phenomena are then discretized using a Finite Volume method and the subdomains are coupled by a nonoverlapping alternating method with interface relaxation [QV91].

The paper is organized as follows. We first review the mathematical formulation of the physical phenomena that are taken into account. Next the discretization and the DD method that have been chosen are presented. Finally, numerical results are shown.

[1] Ceres is a joint project between the oil companies Amoco, British Petroleum, Elf, and the Institut Français du Pétrole.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 Governing Equations

At present, we consider only incompressible one-phase (water) fluid flow in a 2D section of a sedimentary basin. The set of partial differential equations that govern the phenomena is the following one — subscripts w and s designate the water phase and the solid phase, respectively.

Mass conservation of solid and of water are written as:

$$\begin{aligned} \frac{\partial}{\partial t}\Big(\rho_s(1-\phi)\Big) + \mathrm{div}\Big(\rho_s(1-\phi)\overrightarrow{V}_s\Big) &= q_s, && (2.1)\\ \frac{\partial}{\partial t}(\rho_w\phi) + \mathrm{div}(\rho_w\phi\overrightarrow{V}_w) &= q_w, && (2.2) \end{aligned}$$

where ρ_α is the density of phase α, ϕ is the porosity of the medium, $\overrightarrow{V}_\alpha$ is the velocity of phase α, q_α the quantity of phase α deposited on the top of the basin during sedimentation (or removed during erosion).

Darcy's law is written as:

$$\overrightarrow{U_w} = \phi(\overrightarrow{V_w} - \overrightarrow{V_s}) = -\frac{\overline{\overline{K}}}{\mu}\Big(\overrightarrow{gradP} - \rho_w\overrightarrow{g}\Big), \tag{2.3}$$

where P is the pressure of water, $\overrightarrow{g}$ is the gravitational acceleration vector, μ_w the viscosity of water. $\overline{\overline{K}}$ is the intrinsic permeability tensor of the porous medium, and depends heavily on the lithology under consideration. It can vary by several orders of magnitude — up to four — from one layer to the other.

Compaction of the porous medium is supposed to be merely vertical (the horizontal component of the solid velocity is zero). Mechanical equilibrium is then written as:

$$\frac{\partial\sigma}{\partial z} = (\phi\rho_w + (1-\phi)\rho_s)g, \tag{2.4}$$

where σ is the total vertical stress. Compaction is described by the following rheology:

$$\phi = \phi^o exp(-(\sigma - P)/\sigma_e^o). \tag{2.5}$$

The problem is therefore defined by a system whose main unknowns are ϕ, P, σ, $V_{s,z}$ and $\overrightarrow{U_w}$. The principal equations are the two conservation laws, the rheology, the mechanical equilibrium and Darcy's equations. Introducing (2.3) and (2.5) in (2.2) show that the equations are nonlinear parabolic with respect to the pressure P.

3 Discretization

Although the model we are considering is rather simple, we want to choose a method that can be easily extended to more complex fluid flow model and especially compressible three-phase flow. Therefore, the method chosen to discretize the conservation equations is a Finite Volume method [FWS96].

We give in this section, first the main characteristics of the discretization, then some details about the water conservation equation discretization and especially the flux approximation. In this first stage of the project, we consider only subdomains which are separated by vertical interfaces and as compaction is vertical, the coupling between two subdomains only affects the water conservation equation, that is to say, the flux approximation.

Main Characteristics

A grid is chosen that follows the stratigraphic layers. Each cell is homogeneous and the boundaries between two layers correspond to a series of interfaces of adjacent cells. A cell is a quadrangle whose vertices are located along vertical lines. As the grid deforms with the solid skeleton, vertical compaction constrains cell vertices to move only vertically and thus to remain along vertical lines.

The discretization methods are listed in the Table 1, where the discrete unknowns are also specified. We denote by the subscript k an unknown located at the center of the cell Ω_k and by the subscript δ an unknown associated to a vertical edge δ. P_δ is an auxiliary unknown which is given by the flux continuity on each vertical edge as a function of the pressures in the neighboring cells. At each time step, the discretized equations form a system which is nonlinear for the discrete unknowns ϕ_k, P_k, σ_δ, ϕ_δ. It is solved using a Newton method where at each iteration, the system reduces to a linear system in the unknowns P_k, σ_δ.

Table 1 Discretization methods and discrete unknowns.

Equation	Discretization	Discrete unknown
Solid Conservation (2.1)	Finite Volume	ϕ_k
Water Conservation (2.2)	Finite Volume	P_k
Mechanical Equilibrium (2.4)	Finite Difference	σ_δ
Rheology (2.5)		ϕ_δ
Darcy's law and Flux continuity	Finite Difference	P_δ (auxiliary unknown)

Flux Approximation

Following the Finite Volume principle [EGHon], the water conservation equation is integrated over each cell Ω_k. As the cell evolves at the velocity $\vec{V}_s$, it gives:

$$\frac{d}{dt}\int_{\Omega_k} \phi d\omega + \sum_{\delta \subset \delta\Omega_k} \int_\delta \vec{U}_w \cdot \vec{n} d\gamma = \int_{\Omega_k} q_w d\omega \tag{3.6}$$

where $\vec{n}$ is the outward normal to cell Ω_k and $\vec{U}_w$ is given by (2.3).

In order to give some details about the flux approximation, we assume for the sake of simplicity that the mesh is composed of rectangles and that the permeability tensor is diagonal. We denote by K_s and K_a the two diagonal coefficients of $\bar{\bar{K}}$ and by F_δ the flux approximation $-l_\delta \frac{\bar{\bar{K}}}{\mu}(\overrightarrow{gradP} - \rho_w \vec{g}).\vec{n}$, where l_δ is the edge length. Let us distinguish the case where the edge is located inside a subdomain and the case where it is on the boundary:

- *Edge located inside a subdomain*
We consider an edge δ and Ω_a, Ω_b its two adjacent cells (see Fig. 1(a)). P_a

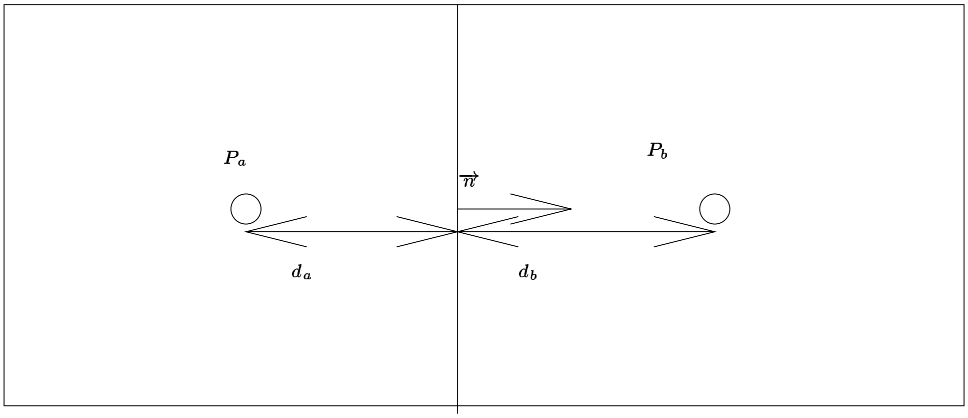
and P_b are the unknown pressures in cells Ω_a, Ω_b. Taking into account the eventual discontinuity of K_s from cell Ω_a to cell Ω_b and expressing the flux continuity on edge δ leads to the following approximation [FWS96]:

$$F_\delta = -l_\delta \overline{\frac{K_s}{\mu}} \frac{P_b - P_a}{d_b + d_a},$$

where $\overline{\frac{K_s}{\mu}}$ is the harmonic mean of $\frac{K_s}{\mu}$ weighted by the distances d_a and d_b.

- *Edge located on the boundary of a subdomain*
  We consider an edge δ located on the boundary of the subdomain, Ω_a its adjacent cell and P_δ the pressure at the center of δ (see Fig. 1(b)).
  If a Dirichlet boundary condition is set on δ, the value of P_δ is given by the boundary condition and the flux is approximated by:
  $$F_\delta = -l_\delta \left(\frac{K_s}{\mu}\right)_a \frac{P_\delta - P_a}{d_a}.$$
  If a Neumann boundary condition is set on δ, the value of F_δ is given by the boundary condition and the pressure is approximated by:
  $$P_\delta = -\frac{d_a}{l_\delta} \left(\frac{\mu}{K_s}\right)_a F_\delta + P_a.$$

Figure 1 (a) Edge inside a subdomain, (b) Edge on the boundary.



(a)

4 Domain Decomposition Method

As for the discretization method, we want to implement a DD method that can be easily extended to more complex model. Moreover, even for the model considered here, the set of equations is non linear. Therefore, we have chosen a method which does not require too many properties of the equations such as linearity, symmetry, etc. This method is nonoverlapping DD method with Dirichlet-Neumann sweep and interface relaxation of the Dirichlet condition as suggested in [QV91]. It is directly applied to the nonlinear system of discretized equations. We describe first the main steps of the algorithm, then the choice of the relaxation parameter.

Algorithm

Let us consider a basin divided into nonoverlapping subdomains Ω_1 and Ω_2 and denote by Γ the interface between the two subdomains. It can be shown that the problem on the global domain is equivalent to the problem on the two subdomains if and only if pressure and flux are continuous through the interface. Once discretized, the global domain and subdomains problems are equivalent if and only if

$$\forall \delta \subset \Gamma, \quad P^1_\delta = P^2_\delta \text{ and } F^1_\delta = -F^2_\delta,$$

where the superscript i refers to a quantity related to subdomain Ω_i for $i = 1, 2$.

The algorithm takes the values of the pressure on the interface as main unknowns: λ_δ, $\delta \subset \Gamma$. It is an iterative algorithm in which, for each iteration k, the following stages are executed:

- Solve $\begin{cases} \text{PDE system in } \Omega_1 \\ \text{BC on } \delta\Omega_1 \backslash \Gamma \\ P^1_\delta = \lambda^k_\delta, \ \forall \delta \subset \Gamma \end{cases}$ then solve $\begin{cases} \text{PDE system in } \Omega_2 \\ \text{BC on } \delta\Omega_2 \backslash \Gamma \\ F^2_\delta = -F^1_\delta, \ \forall \delta \subset \Gamma \end{cases}$
- Update λ: $\lambda^{k+1}_\delta = (1-\theta)\lambda^k_\delta + \theta P^2_\delta \quad \forall \delta \subset \Gamma$,
 where θ is the relaxation parameter.

Relaxation Parameter

As shown by the numerical tests (see Section 5), the optimal value of the relaxation parameter, i.e., the value for which the number of iterations is minimal, strongly depends on the basin characteristics (permeability heterogeneities). Therefore, it is necessary to implement an algorithm which automatically computes this parameter. The algorithm chosen here is the one suggested by A. Quarteroni and presented in [HK92], which computes the relaxation parameter θ^k at each iteration k in the following way. Defining error functions:

$$\begin{aligned} e^{1,k}_\delta &= P^{1,k}_\delta - P^{1,k-1}_\delta, \quad e^{2,k}_\delta = P^{2,k}_\delta - P^{2,k-1}_\delta, \\ z^k_\delta(\theta) &= \theta P^{2,k}_\delta + (1-\theta)P^{1,k}_\delta \quad \text{for} \quad \delta \subset \Gamma, \end{aligned}$$

the unique number which minimizes $||z^k(\theta) - z^{k-1}(\theta)||^2 = \sum_{\delta\subset\Gamma}(z^k_\delta(\theta) - z^{k-1}_\delta(\theta))^2$ is

$$\theta^k_{opt} = \frac{\sum_{\delta\subset\Gamma} e^{1,k}_\delta (e^{1,k}_\delta - e^{2,k}_\delta)}{\sum_{\delta\subset\Gamma}(e^{1,k}_\delta - e^{2,k}_\delta)^2}.$$

5 Numerical Results

The results presented here have been obtained for a basin which is already deposited and that is compacting under the effect of a vertical stress applied at the top of the basin. We first consider basins with a simple lithologic composition for which we study the behavior of the DD method. Then, we are interested in more complex basins divided in several subdomains.

Simple Basins

Two basins of 10 layers, divided in 2 subdomains (10 interface edges), are considered:

- a homogeneous basin only composed of shales, that is to say, of impervious sediments.
- a basin composed of two homogeneous subdomains; the first one is made of shales and the second one of sandstones, that is to say, of pervious sediments. There are four orders of magnitude between the permeability in the two subdomains. The shales subdomain is the one on which a Dirichlet boundary condition is set.

The results, in terms of the maximum number of iterations needed during the different time steps of the simulation, are presented in Table 2. A star indicates that the algorithm does not converge. Each column gives the results for a certain value of the relaxation parameter. When a real value is written, it means that this value has been kept constant during all the simulation. The column corresponding to θ_{opt} gives the results obtained with the algorithm presented in the previous section. For these simple cases, a satisfying value of θ would have been 0.5. The next set of results shows that this is no longer the case for complex basins. Moreover, the results obtained with the dynamical computation of θ are also very good. For the second basin, it should however be noticed that, if a Dirichlet boundary condition is set on the sandstones domain, the DD algorithm does not converge, and this, even for very small value of θ. The Neumann boundary condition has to be set on the more pervious subdomain.

Table 2 Simple basins: number of iterations for different values of θ.

θ	1	0.5	0.25	0.05	θ_{opt}
Homogeneous Shales	*	3	17	117	4
Shales-Sandstones	2	19	42	210	4

Complex Basins

We now consider more complex basins composed of blocks separated by faults. Blocks and faults are computational subdomains and there is only one column of cells in each fault. The blocks consist in alternated shales and sandstones layers while the faults are made of sandstones. The following basins are considered:

- 2 blocks of 5 layers, separated by 1 fault (3 subdomains, 10 interface edges)
- 2 blocks of 10 layers, separated by 1 fault (3 subdomains, 20 interface edges)
- 3 blocks of 10 layers, separated by 2 faults (5 subdomains, 40 interface edges)
- 3 blocks of 20 layers, separated by 2 faults (5 subdomains, 80 interface edges) This last basin is represented in Figure 2.

For all these tests, Neumann boundary conditions have been set on the two boundaries of each fault. The results are presented in Table 3, in the same way as in Table 2. The evolution of the relaxation parameter and of the corresponding error for the basin of Figure 2 is represented on Figure 3. The results show that the dynamical computation of the relaxation parameter represents an important gain in the number of iterations compared to a constant value of θ. Moreover, the number of iterations increases slowly with the number of subdomains and of interface unknowns.

6 Conclusion

A DD method has been implemented for modeling one-phase flow in a sedimentary basin. The equations are discretized by a Finite Volume method and the DD method is a nonoverlapping DD method with Dirichlet-Neumann sweep and interface relaxation. This method gives satisfactory results for simple and rather complex basins. It is currently being extended to non-matching meshes and to three-phase fluid flow.

Table 3 Maximum number of iterations for different values of θ.

θ	0.25	0.05	θ_{opt}
5 layers, 3 subdomains	25	137	7
10 layers, 3 subdomains	26	144	10
10 layers, 5 subdomains	*	125	16
20 layers, 5 subdomains	*	126	23

Acknowledgement

We would like to thank Amoco, British Petroleum, and Elf for authorizing the publication of these results. We are very grateful to Pr. Galloüet and Pr. Glowinski for helpful discussions and to E. Flauraud who completed the major part of the software development.

REFERENCES

[EGHon] Eymard R., Gallouet T., and Herbin R. (in preparation) *Finite Volume*

Figure 2 Complex basin: 3 blocks of alternated shales and sandstones layers separated by 2 faults of sandstones.

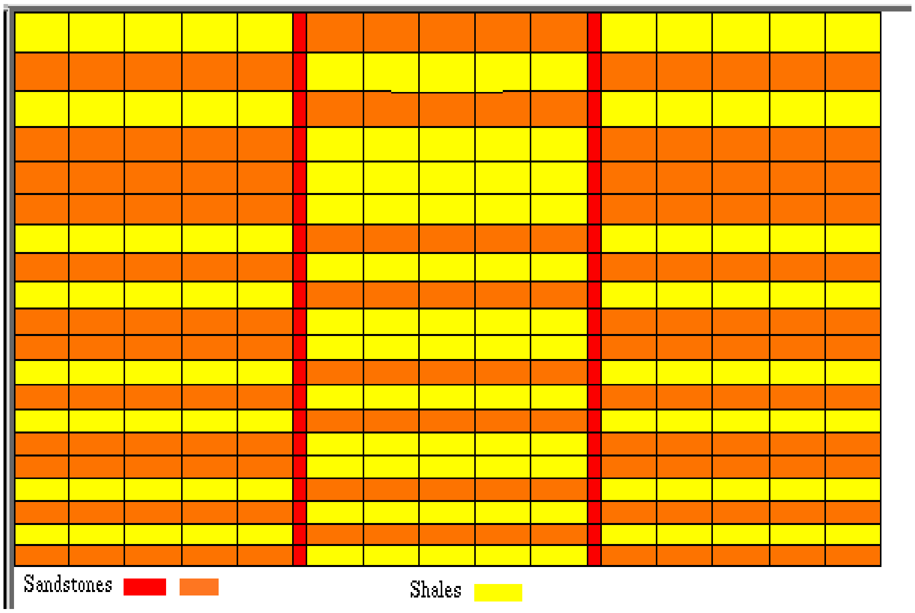
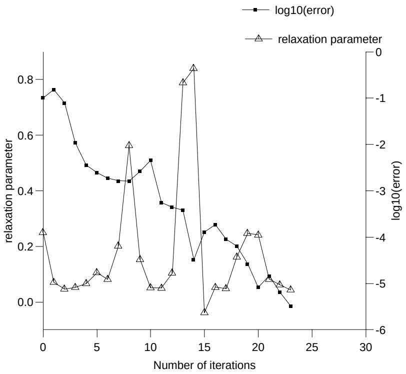


Figure 3 Evolution of the relaxation parameter and of the corresponding error during one time step of the simulation for the basin shown on Fig. 2.



Methods. Handbook of Numerical Analysis. P.G. Ciarlet and J.L. eds, North-Holland.

[FWS96] Faille I., Wolf S., and Schneider F. (August 1996) Finite volume and finite element methods in basin modeling. Report 43110, Institut Français du Pétrole, Rueil Malmaison, France.

[HK92] Henderson R. and Karniadakis G. E. (1992) Hybrid spectral element methods for flows over rough walls. In Keyes D. E., Chan T. F., Meurant G. A., Scroggs J. S., and Voigt R. G. (eds) *Proc. Fifth Int. Conf. on Domain Decomposition Meths.*, pages 485–497. SIAM, Philadelphia.

[QV91] Quarteroni A. and Valli A. (1991) Theory and application of steklov-poincaré operators for boundary-value problems. *R. Spiegler, Applied and Industrial Mathematics* pages 179–203.

[UBD$^+$90] Ungerer P., Burrus J., Doligez B., Chnet P. Y., and Bessis F. (1990) Basin evaluation by integrated two–dimensional modeling of heat transfer, fluid flow, hydrocarbon generation, and migration. *AAPG Bulletin* 74(3): 309–335.

87

Multilevel Adaptive Methods for Semilinear Equations with Applications to Device Modelling

R. C. Ferguson and I. G. Graham

1 Introduction

The drift-diffusion equations modelling the steady state electrical behaviour of a semiconductor device present several challenging problems for the numerical analyst. These equations form a 3 × 3 coupled elliptic system with one or more small parameters and are typically subject to mixed boundary conditions on non-smooth domains. The solutions of this system contain both interior layers and geometric boundary singularities which require appropriately graded meshes for their accurate approximation. Since these irregularities are very complex and the precise position of interior layers is quite a delicate matter ([MRS90]), it is not possible to derive suitable meshes *a priori* and a mesh refinement process based on *a posteriori* error estimation is essential for adequate resolution. A variety of approaches to adaptivity in device modelling can be found in the numerical engineering literature (e.g., [KR93, BCD92]). Much of this is based on heuristics, e.g., refinement based on doping profile. Here we derive rigorous error estimates for a reduced class of problems and a theoretically justified efficient method of implementation.

At least two difficulties have to be considered. The first is the construction of an error estimator which works well even in the presence of small parameters. The second stems from the highly nonlinear nature of the system: Each nonlinear solve requires many linear solves which form the computational core of the solution process. If a mesh is to be adaptively determined, then in principle one may be faced with solving the nonlinear system on several intermediate meshes. To reduce the cost of such a process one should in principle solve the intermediate problems up to an accuracy commensurate with the quality of those meshes, and compute accurate solutions only on the most accurate meshes.

In this paper we shall survey some recent results on the resolution of these two

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

difficulties in the context of the single semilinear equation:

$$-\lambda^2 \Delta u + f(u) = 0 \tag{1.1}$$

on a polygonal domain $\Omega \subset I\!R^2$ subject to mixed boundary conditions $u = g$ on $\partial\Omega_D$, $\partial u/\partial n = 0$ on $\partial\Omega_N$, where $\partial\Omega_D$ and $\partial\Omega_N$ partition $\partial\Omega$. The "Gummel iteration" for the semiconductor system can be written as sequences of such semilinear scalar problems, but including some where the second-order term $-\lambda^2\Delta u$ is replaced by a operator of the form $-\nabla.a\nabla u$, with rapidly varying coefficient function a. A detailed analysis of (1.1) is thus the first step in the design of a fully adaptive device model. In fact, in its "off" state, the electrostatic potential u of the device satisfies the equation

$$-\lambda^2 \Delta u + 2\delta^2 \sinh u - d = 0. \tag{1.2}$$

Here λ^2, δ^2 can both be small and the doping profile d satisfies $|d| \leq 1$ but varies in sign across interfaces interior to Ω. On the Dirichlet boundary $\partial\Omega_D$, u is required to satisfy $u = \sinh^{-1}\left(d/2\delta^2\right)$.

In this paper we present some a posteriori error estimates for (1.1) which work well under extreme parameter ranges and in the presence of geometric singularities. For the practical implementation of the refinement process we propose an inexact Newton method, related to those in [AXE93] and [XU94], which solves (1.1) by resolving the nonlinearity on a coarse mesh and then computing a sequence of corrections by solving linear problems on successively finer grids. Numerical experiments show that this method is capable of reproducing qualitative features of solutions of (1.2) (known from singular perturbation theory), by using considerably fewer linear iterations than those used in solving (1.2) to full accuracy at each refinement step. Full details of the results reviewed here are in the thesis [FER97].

2 A Posteriori Error Estimates for Semilinear Equations

Consider the problem (1.1) subject to the stated boundary conditions. Assume there exists a weak solution $u_0 \in \mathrm{L}_\infty(\Omega)$ with $\Delta u_0 \in L_\infty(\Omega)$, that f has two continuous derivatives on $I\!R$, that $g \in \mathrm{H}^{\frac{3}{2}}(\partial\Omega_D)$ and that λ is some small parameter. With these assumptions it is shown in [FER97] (using well known linear results such as [GRI92]) that $u_0 \in \mathrm{H}^{1+\alpha}(\Omega)$, where $\alpha \in (1/4, 1]$ is a fixed constant, depending purely on the interior angles of Ω at points where the boundary segments meet. It is also assumed that the Fréchet derivative of the operator in (1.1) evaluated at u_0 has a bounded inverse as an operator from $\mathrm{H}^1_{0,\partial\Omega_D}$ to $(\mathrm{H}^1)'$.

Define a shape regular triangulation $\mathcal{T}_h$ of Ω, whose union is Ω. For each triangle $T_k \in \mathcal{T}_h$ define h_k to be its diameter and let $\mathcal{E}_h$ denote the set of edges of the triangles. h_τ is defined to be the length of an edge $\tau \in \mathcal{E}_h$. If $h, \underline{h}$ are the maximum and minimum triangle diameters we require the very mild assumption that $h\log(1/\underline{h})^{\frac{1}{2}} \to 0$, as $h \to 0$. Then, for h sufficiently small, there is a finite element solution, u_h, of (1.1) which is unique in a ball centered on u_0 in H^1. Let $[\partial u_h/\partial n]_\tau$ be the difference in the normal derivative of u_h across an edge τ of a triangle. Then for constants C_1 and C_2, the H^1 and L_2 *a posteriori* error estimates may be written as:

$$\|u_0 - u_h\|_{\mathrm{H}^1} \leq C_1 \left[\lambda^2 \left\{ \sum_{\tau \in \mathcal{E}_h} h_\tau^2 \left[\frac{\partial u_h}{\partial n} \right]_\tau^2 \right\}^{\frac{1}{2}} + \left\{ \sum_{T_k \in \mathcal{T}_h} \|h_k f(u_h)\|^2_{\mathrm{L}_2(T_k)} \right\}^{\frac{1}{2}} \right], \quad (2.3)$$

$$\|u_0 - u_h\|_{\mathrm{L}_2} \leq C_2 \left[\left\{ \sum_{T_k \in \mathcal{T}_h} h_k^{2\alpha} \|u_0 - u_h\|^2_{\mathrm{H}^1(T_k)} \right\}^{\frac{1}{2}} + \|u_0 - u_h\|^2_{\mathrm{H}^1} \right]. \quad (2.4)$$

The estimates (2.3) and (2.4) are analogous to estimates in [VER94] and [VER96]. However, in [VER96], the loss of H^2 regularity due to reentrant corners and/or mixed boundary conditions is handled by use of a scale of W_p^1 spaces with variable p. In this work we instead use the scale $\mathrm{H}^{1+\alpha} = \mathrm{W}_2^{1+\alpha}$. Similar L_2 estimates, but assuming full H^2 regularity, are found in [EEHJ95].

Our adaptive scheme will use the L_2 *a posteriori* error estimate (2.4). In it the second term on the right hand side is estimated using (2.3) and the first term is estimated by assuming that $\|u_0 - u_h\|_{\mathrm{H}^1(T_k)}$ may be estimated by the contribution to the right hand side of (2.3) from T_k. The constants, C_1 and C_2, are estimated in [FER97] and the best theoretical bounds are of order λ^{-2} in general. However in [FER97] it is also shown heuristically that even for small λ the numerical values of C_1 and C_2 are likely to grow more slowly than this. In order to ensure that our adaptive process is robust with respect to λ we estimate C_1 and C_2 by extrapolation from computed error estimates (2.4) for each pair of successive triangulations. The computed C_1 and C_2 conform to the heuristics mentioned above. Our adaptive scheme is: Choose an initial coarse triangulation and a tolerance. Then:

- Calculate the current finite element solution to the problem.
- Calculate the *a posteriori* error estimate (2.4), after having estimated the constants C_1, C_2. (On the first refinement step these are arbitrarily chosen to be 1.)
- If the error is greater than the chosen tolerance then refine the triangulation: A triangle is refined if its contribution to the total *a posteriori* error estimate exceeds the average error over the triangles by some tolerance.
- Repeat until the tolerance is achieved.

To test the adaptive scheme consider the "off" state PN diode problem: seek u satisfying (1.2) subject to $u = \sinh^{-1}(d/2\delta^2)$ on $\partial\Omega_D$ and $\partial u/\partial n = 0$ on $\partial\Omega_N$. Here Ω is the unit square and the boundary $\partial\Omega$ is split into $\partial\Omega_D = \{0 \times [0, 1/2)\} \cup \{1 \times [0, 1]\}$ and $\partial\Omega_N = \partial\Omega \backslash \partial\Omega_D$. d in (1.2) is the piecewise constant doping profile of the device and takes a value of $+1$ in the region $\{(x, y) : x^2 + y^2 \leq 0.25\}$ and -1 elsewhere. λ and δ are small parameters, which depend on various physical attributes. In this experiment λ will vary, but δ^2 is fixed at 1×10^{-7}. For this problem $u_0 \in \mathrm{H}^{1+\alpha}$, where $\alpha < 1/2$. (In the experiments we used $\alpha = 1/2$ in (2.4)).

It has been shown using singular perturbation theory, [MAR84], that the solution of this problem has a layer at the interface between $d = +1$ and $d = -1$ and the width of this layer is of order $\lambda|\log \lambda|$ as $\lambda \to 0$. To test our adaptive scheme we try to capture the correct order of λ in the width of the interior layer in the computed finite element solution as λ varies. In principle it is difficult to define where a layer "begins" and "ends". However in this case it is known that outside the layer the exact solution is "flat" and in the regions where $d = \pm 1$ it has essentially the values $u = \pm \sinh^{-1}(1/2\delta^2)$. The layer is defined to start and end when the finite element solution is bounded away from these values by a small number — here we choose 0.03. Selected results for the PN diode problem are presented in Table 1. These show that the desired order of λ is present in the computed widths. We also observe that the number of nodes needed to compute successively more severe layers does not blow up. All the experiments in this paper are obtained using a program combining the packages PETSc (Argonne National Laboratory) and Femlab (Chalmers University of Technology) and use a tolerance of 5×10^{-3} for the adaptivity.

Table 1 shows how the numerically computed width of the layer depends on λ as $\lambda \to 0+$. The theory predicts that the width is of order $\lambda \log(\lambda)$ as $\lambda \to 0+$.

λ^2	Size of initial grid	Number of refinements	Final number of nodes	Width of layer	Order of λ in width
1×10^{-4}	10 × 10	15	2963	0.1527	—
5×10^{-5}	10 × 10	12	3894	0.1111	0.92
1×10^{-5}	20 × 20	16	6453	0.0526	0.93
5×10^{-6}	20 × 20	12	3667	0.0382	0.93
1×10^{-6}	30 × 30	10	4166	0.0193	0.85

3 The Inexact Newton Method

The adaptive scheme described in the previous section solves, to full accuracy, a nonlinear system for each triangulation before computing error estimates and refining the grid. Since a typical refinement process can involve refining a number of triangulations, this may involve a lot of unnecessary effort. In this section we propose an adaptive scheme that considerably reduces this effort. The scheme is similar to those proposed in Xu [XU94] and Axlesson [AXE93].

Our inexact Newton method proceeds by solving the nonlinear problem, to full accuracy, on an initial coarse triangulation and then computes corrections to the calculated solution on a sequence of successively finer triangulations. These corrections involve solving one linearised problem on each of the finer triangulations.

For this adaptive procedure it is rather difficult to prove *a priori* convergence. Instead we justify the scheme theoretically *under the assumption* that a sequence of triangulations of optimal approximation power are being generated (only weak assumptions avoiding quasi-uniformity are imposed on these meshes). Under these assumptions we can prove the well-posedness and convergence of the inexact Newton method. In Section 4 we shall demonstrate, empirically, the effectiveness of the adaptive variant of this method.

Thus, for the theory, suppose that we have a sequence of shape regular triangulations

Table 2 The number of linear solves required to solve the PIN diode problem for different values of λ and δ using the two methods.

λ^2	δ^2	Linear solves for meth. of Section 2	Linear solves for inexact Newton
1×10^{-4}	1×10^{-5}	32	17
1×10^{-4}	1×10^{-7}	22	17
1×10^{-4}	1×10^{-8}	66	53
1×10^{-5}	1×10^{-4}	64	20
1×10^{-7}	1×10^{-4}	156	32
1×10^{-8}	1×10^{-4}	310	14

$\{\mathcal{T}_h^k\}$, define h^k to be the maximum diameter of the triangles in $\mathcal{T}_h^k$ and denote $\mathcal{V}_h^k$ to be the piecewise linear finite element space corresponding to $\mathcal{T}_h^k$. Consider problem (1.1). The finite element discretisation on the kth triangulation induces a map $F_h^k : \mathcal{V}_h^k \to (\mathcal{V}_h^k)'$ defined by $(F_h^k(u_h), v_h) = (\nabla u_h, \nabla v_h) + (f(u_h), v_h)$, which has the linearisation $(F_h^k)' : \mathcal{V}_h^k \to L(\mathcal{V}_h^k, (\mathcal{V}_h^k)')$ [where $L(A, B')$ denotes the set of all linear operators $A \to B'$ and B' denotes the dual space of B]. In each case $\mathcal{V}_h^k$ must be supplied with appropriate boundary conditions. Then if u_h^k is the true finite element solution on the kth triangulation, the inexact Newton scheme generates a sequence $\{\hat{u}_h^k\}$ defined by the algorithm:

1. Set $\hat{u}_h^0 = u_h^0$, the exact solution of the nonlinear finite element problem $F_h^0(u_h^0) = 0$ in $\left(\mathcal{V}_h^0\right)'$.
2. For $k = 0, 1, 2, \ldots$, iterate the two steps:
   - Solve for $\hat{e}_h^{k+1} \in \mathcal{V}_h^{k+1}$: $\left(F_h^{k+1}\right)' (\hat{u}_h^k)\; \hat{e}_h^{k+1} = -F_h^{k+1}(\hat{u}_h^k)$
   - Update $\hat{u}_h^k$: $\hat{u}_h^{k+1} = \hat{u}_h^k + \hat{e}_h^{k+1}$

Define $\Pi_h^k u_0$ to be the finite element interpolant of u_0 at the nodes of the triangulation $\mathcal{T}_h^k$. We assume, for all k, that the following approximation properties ([STW90]) hold: $\|u_0 - \Pi_h^k u_0\|_{\mathrm{H}^1} \le C h^k \|u_0\|_{\mathrm{H}^2_W}$, $\|u_0 - \Pi_h^k u_0\|_{\mathrm{L}_2} \le C(h^k)^2 \|u_0\|_{\mathrm{H}^2_W}$ and $\|u_0 - \Pi_h^k u_0\|_{\mathrm{L}_\infty} \le C(h^k)^2 \|u_0\|_{\mathcal{C}^2_W}$. Here H^2_W is a weighted H^2 Sobolev space with weight decaying sufficiently quickly near the points of singularities on the boundary, $\mathcal{C}^2_W$ is an analogous weighted $\mathcal{C}^2$-space. It is a standard result that, if h^0 is sufficiently small, the true finite element solution on the kth triangulation, u_h^k, exists and satisfies the *a priori* error estimate:

$$\|u_0 - u_h^k\|_{\mathrm{H}^1} \le C_3 h^k \|u_0\|_{\mathrm{H}^2_W}, \quad \text{for all } k, \tag{3.5}$$

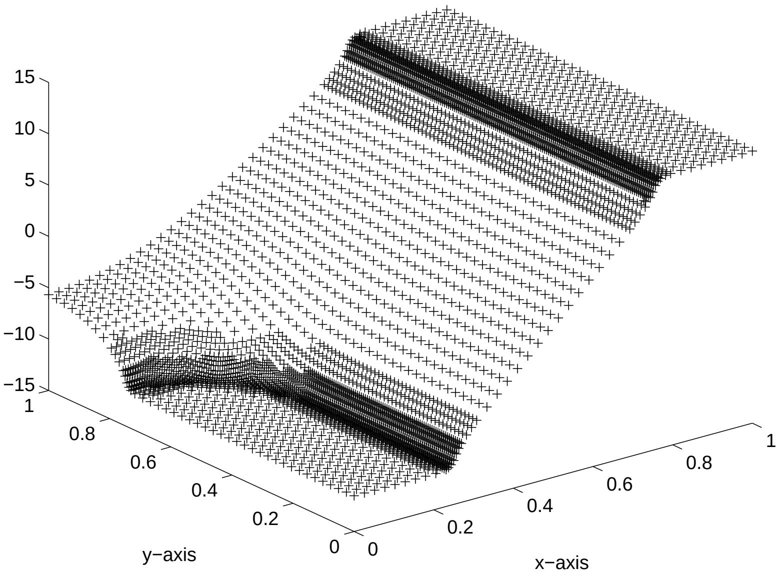
where C_3 is a constant independent of h^k and k.

In order to prove that the inexact Newton method is well-defined we need to assume that the triangulations are not too severely refined at each step. This is natural since it essentially ensures that the sequence of inexact Newton iterates stays within some suitably small ball centered on the true solution. Thus we assume that there exists a $\gamma > 0$ and $\varepsilon \in (0, 1)$, independent of h and k, such that for all k $(h^k)^2 \leq \gamma(h^0)^{1-\varepsilon}h^{k+1}$. Then it has been proved in [FER97] that, for h^0 sufficiently small, the inexact Newton solution on the kth triangulation, $\hat{u}_h^k$, is well defined for all k and satisfies the error estimate:

$$\|u_0 - \hat{u}_h^k\|_{\mathrm{H}^1} \leq C_3(1 + C_4(h^k)^{\varepsilon})h^k\|u_0\|_{\mathrm{H}^2_W}. \tag{3.6}$$

C_4 is a constant independent of h^k and k, and C_3 is the constant appearing in (3.5). Thus, neglecting higher order terms, the *a priori* error estimate for $\hat{u}_h^k$ is identical to that for u_h^k. It can also be shown that, apart from perturbations of order $(h^k)^{1+\epsilon}$, $\|u_0 - \hat{u}_h^k\|_{\mathrm{H}^1}$ is bounded above and below by $\|u_0 - u_h^k\|_{\mathrm{H}^1}$.

Figure 1 The defect correction finite element solution to the PIN diode problem when $\delta^2 = 1 \times 10^{-5}$ and $\lambda^2 = 1 \times 10^{-4}$.

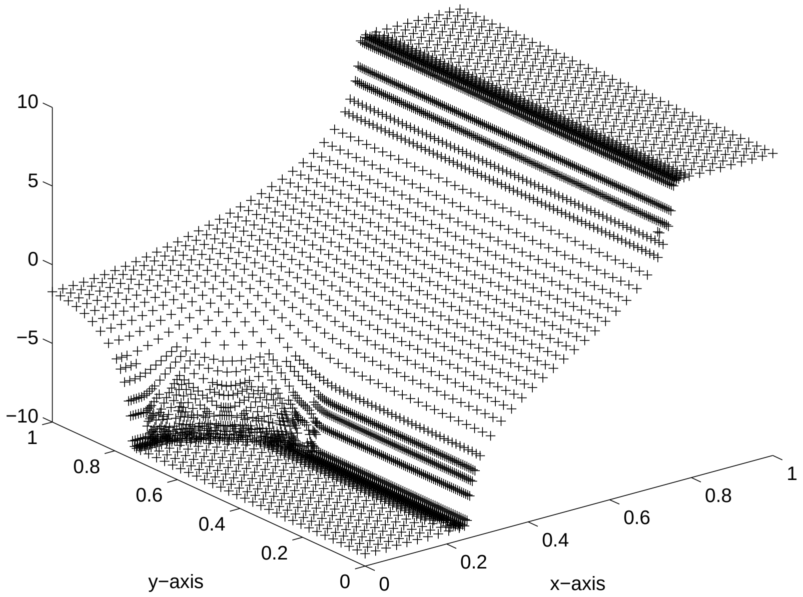


4 Experiments with the Adaptive Inexact Newton Method

The error estimates in the previous section are obtained with *a priori* determined triangulations which have optimal interpolation properties. In practice triangulations determined using adaptive L_2 refinement are used.

To test the inexact Newton method consider the PIN diode problem in its "off" state. This is a problem of the form (1.2) with the mixed boundary conditions considered in Section 2, where Ω is taken to be the unit square and $d = +1$

Figure 2 The defect correction finite element solution to the PIN diode problem when $\delta^2 = 1 \times 10^{-4}$ and $\lambda^2 = 1 \times 10^{-5}$.



in the set $\Omega_+ = \{(x,y) : \; 0.75 \le x \le 1, \; 0 \le y \le 1\}$, $d = -1$ in the set $\Omega_- = \{(x,y) : \; 0 \le x \le 0.25, \; 0 \le y \le 0.5 \text{ or } \sqrt{x^2 + (y-0.5)^2} \le 0.25\}$ and $d = 0$ in $\Omega_0 = \Omega \backslash (\overline{\Omega}_+ \cup \overline{\Omega}_-)$. The Dirichlet boundary, $\partial\Omega_D$, is the set $\{0 \times [0, 1/2)\} \cup \{1 \times [0,1]\}$.

It has been shown in [MRS90] that the solution to the PIN diode problem has substantially different asymptotic behaviour in each of the cases $\lambda \ll \delta \to 0$ and $\delta \ll \lambda \to 0$. In the former case,

$$\psi|_{\Omega_+} = \sinh^{-1}\left(\frac{1}{2\delta^2}\right), \; \psi|_{\Omega_-} = \sinh^{-1}\left(\frac{-1}{2\delta^2}\right) \text{ and } \psi|_{\Omega_0} = 0,$$

whereas in the latter,

$$\psi|_{\Omega_+} = \sinh^{-1}\left(\frac{1}{2\delta^2}\right), \; \psi|_{\Omega_-} = \sinh^{-1}\left(\frac{-1}{2\delta^2}\right) \text{ and } \Delta\psi = 0 \text{ in } \Omega_0.$$

We use these known asymptotics to test the accuracy and efficiency of the adaptive inexact Newton method for a variety of λ and δ. The initial coarse triangulation was refined using the L_2 error estimate (2.4) as described in the previous section. To satisfy the conditions that the triangulations should not change too much, a maximum of 10% of the triangles were refined at each iteration and a triangle was only refined if its error estimate was twice the average of all the error estimates. Pictures of two solutions produced using the inexact adaptive Newton scheme are presented in Figures 1 and 2. These show the correct asymptotic form above (more details are in [FER97]).

The aim of the inexact Newton method is to reduce the amount of computational effort needed to find accurate finite element solutions. The method introduced in Section 2 solves a nonlinear problem for each triangulation, whereas the inexact Newton method only requires one nonlinear solve on the coarsest triangulation and then a linear solve for each of the fine triangulations. The number of linear solves required for each method for a variety of λ and δ is presented in Table 2. It was found that if the inexact Newton scheme was started with too coarse an initial triangulation or the triangles were refined too quickly then the iteration diverged. Even though the inexact Newton method may use a larger number of triangulations and nodes than the method in Section 2 [since the grids are refined more cautiously], we found that it still requires significantly fewer linear solves to produce solutions of the same accuracy.

REFERENCES

[AXE93] AXELSSON O. (1993) On mesh independence and Newton-type methods. *Applications of Mathematics* 38: 249–265.

[BCD92] BACCUS B., COLLARD D., and DUBOIS E. (1992) Adaptive mesh refinement for multilayer process simulation using the finite element method. *IEEE Transactions on Computer-Aided Design* 11: 396–403.

[EEHJ95] ERIKSSON K., ESTEP D., HANSBO P., and JOHNSON C. (1995) Introduction to adaptive methods for differential equations. *Acta Numerica* pages 105 – 158.

[FER97] FERGUSON R. (1997) *Numerical Techniques for the Drift-Diffusion Semiconductor Equations.* PhD thesis, University of Bath, Bath, U.K.

[GRI92] GRISVARD P. (1992) *Singularities in Boundary Value Problems.* Masson/Springer-Verlag.

[KR93] KORNHUBER R. and ROITZSCH R. (1993) Self adaptive finite element simulation of bipolar, strongly reverse-biased pn-junctions. *Communications in Numerical Methods in Engineering* 9: 243–250.

[MAR84] MARKOWICH P. (1984) A singular perturbation analysis of the fundamental semiconductor device equations. *SIAM J. Applied Mathematics* 44: 896–928.

[MRS90] MARKOWICH P., RINGHOFER C., and SCHMEISER C. (1990) *Semiconductor Equations.* Springer-Verlag.

[STW90] SCHATZ A., THOMÉE V., and WENDLAND W. (1990) *Mathematical Theory of Finite and Boundary Element Methods.* Birkhäuser.

[VER94] VERFÜRTH R. (1994) A posteriori error estimates for nonlinear problems. Finite element discretization of elliptic equations. *Mathematics of Computation* 62: 445–475.

[VER96] VERFÜRTH R. (1996) A posteriori error estimates for nonlinear problems. L^r-estimates for finite element discretizations of elliptic equations. Technical Report 199, Ruhr-Universität Bochum.

[XU94] XU J. (1994) A novel two-grid method for semilinear elliptic equations. *SIAM J. Scientific Computing* 15: 231–237.

88

Scalability of Industrial Applications on Different Computer Architectures

M. Kahlert, M. Paffrath, and U. Wever

1 Introduction

In the field of partial differential equations, domain decomposition methods are among the most effective parallelization techniques. A direct parallelization leads to so-called data or grid partitioning strategies. The parallel version shows the same algorithmic behavior as the serial one but suffers from a sometimes considerable communication overhead. This disadvantage has led to indirect parallelization strategies which are characterized by an additional outer iteration. This paper presents real life examples from *fabrication of semiconductor devices* and *nuclear reactor analysis*, both for a workstation network and for a multiprocessor-shared memory architecture. In [Kah94] we have shown that for process simulation of semiconductor devices direct parallelization methods are superior to indirect ones, even for workstation clusters. In the present paper we restrict ourselves to direct methods for both application fields: parallel multigrid in connection with Newton's method and parallel nonlinear multigrid. In order to reduce communication overhead, the corresponding smoothing algorithms are changed.

The outline of the paper is as follows: Section 2 illuminates some aspects of multigrid parallelization, Section 3 gives an introduction to the applications, namely point defect and neutron kinetics simulation, and presents results for a workstation cluster and a shared memory system.

2 Aspects of Multigrid Parallelization

For both applications linear prolongation and restriction are used, the restriction operator being the transpose of the prolongation operator. On the finer grids, Gauß-Seidel iterations are used as smoother. On the coarsest grid, the system of equations is solved exactly: by serial Gaussian elimination in case of neutron kinetics simulation,

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

and by a serial or parallel conjugate gradient method in case of process simulation. The serial solution on the coarsest grid can either be computed by a separate host or in parallel by each host. The fastest variant may depend on the underlying hardware architecture. Because communication only takes place in a parallel coarse grid correction and the smoothing algorithm, these algorithms alone are described briefly. After preliminaries on matrix decomposition, parallel conjugate gradients and parallel Gauß-Seidel are described.

Matrix Decomposition

Let Ω, the domain of definition, be decomposed into overlapping or nonoverlapping subdomains $\Omega_i, i = 1, \ldots, N$. Let $Ax = b$ be the basic linear system of equations resulting from, for example, a finite element discretization of the PDE in Ω, A_i the matrix corresponding to the discretization in Ω_i, b_i the right-hand side, and R_i a matrix representing the algebraic restriction of a vector on Ω to the corresponding on Ω_i. Splitting up A_i, b_i and R_i into boundary and inner components, one obtains

$$A_i = \begin{pmatrix} A_i^{II} & A_i^{IB} \\ A_i^{BI} & A_i^{BB} \end{pmatrix}, \quad b_i = \begin{pmatrix} b_i^I \\ b_i^B \end{pmatrix}, \tag{2.1}$$

and

$$R_i = \begin{pmatrix} 0 & \ldots & 0 & I & 0 & \ldots & 0 & 0 \\ 0 & \ldots & & & & \ldots & 0 & R_i^B \end{pmatrix}. \tag{2.2}$$

Using these definitions, a decomposition of the basic system is given by

$$\sum_{i=1}^{N} R_i^T A_i R_i x = \sum_{i=1}^{N} R_i^T b_i. \tag{2.3}$$

Parallel Conjugate Gradient Methods

Conjugate gradient type methods can be parallelized as in [Mey90]. Assume the nonoverlapping domain decomposition with the representation of the basic system given by (2.3). Then two types of local vector representations have to be distinguished:

1. **Non-assembled:** At inner boundaries, a non-assembled vector only contains information on one subdomain and does not coincide with the global vector, e.g., b_i in (2.3).
2. **Assembled:** At inner boundaries an assembled vector coincides with the global vector, e.g., $x_s = R_s x$.

Iteration steps containing summation of two vectors may trivially be parallelized. For a scalar product $< t, r >$ holds

$$< t, r > = < \sum_i R_i^T t_i, r > = \sum_i < t_i, R_i r > = \sum_i < t_i, r_i > \quad ,$$

with the assembled vector r_i and the non-assembled vector t_i in domain i. So $< t, r >$ can be calculated by summing up the local scalar products $< t_i, r_i >$. The norm $\|r\|$

may be calculated in a similar manner. But for stability reasons, $\|r\|$ is calculated by

$$\|r\| = \sqrt{\sum_{i=1}^{N} \|r_i\|_i^2} \quad ; \quad \|r_i\|_i^2 = \sum_{j=1}^{n_i} \alpha_i^j (r_i^j)^2 \quad ,$$

with n_i denoting the number of variables in subdomain i. α_i^j are weights equal to 1 for inner variables and equal to 0 or 1 for variables on subdomain boundaries. For an inner boundary node, α_i^j is 1 for just one adjacent subdomain i_0 and 0 for $i \neq i_0$.

The parallel algorithm is not as stable as the serial one. In practical applications there are sometimes problems in achieving the required accuracy which may have their roots in the assemblies. During the iteration process of the parallel algorithm vectors are assembled with nonzero entries, but the assembled vector should be zero for the converged solution. In the serial program these problems do not occur because the assembly is done before solving the linear system.

Parallel Gauß-Seidel

Parallelization of Gauß-Seidel depends on the type of application, e.g., whether we have a scalar equation or a system of equations. Of course, the numbering of the variables, in particular those at the inner boundaries, plays an important role, also the coupling between variables, and whether we choose standard Gauß-Seidel or Block-Gauß-Seidel. In this section, one parallel variant of standard Gauß-Seidel used in point defect simulation is described with the numbering of variables given by (2.1),(2.2),(2.3).

With a decomposition of matrix A into an upper, lower and a diagonal matrix $(A = U + L + D)$ one iteration of Gauß-Seidel reads:

$$Dx^{k+1} = (b - Lx^{k+1} - Ux^k). \tag{2.4}$$

Applying partition (2.3) of matrix A it holds:

$$\sum_{i=1}^{N} R_i^T D_i R_i x^{k+1} = \sum_{i=1}^{N} R_i^T (b_i - L_i R_i x^{k+1} - U_i R_i x^k). \tag{2.5}$$

Splitting up U_i, L_i, D_i, R_i and b_i into inner and boundary components as in (2.1),(2.2) leads to

$$D_s^{II} x_s^{I,k+1} = b_s^I - L_s^{II} x_s^{I,k+1} - (U_s^{II} x_s^{I,k} + U_s^{IB} x_s^{B,k}) \tag{2.6}$$

for the inner variables of domain s and

$$\sum_{i=1}^{N} R_i^{B,T} D_i^{BB} x_i^{B,k+1} = \sum_{i=1}^{N} R_i^{B,T} (b_i^B - (L_i^{BI} x_i^{I,k+1} + L_i^{BB} x_i^{B,k+1}) - U_i^{BB} x_i^{B,k}) \tag{2.7}$$

for the boundary nodes. (2.7) carries a lot of communication. For some applications it may be therefore advantageous to replace (2.7) by the Jacobi iteration. Applying operator R_s^B one arrives at

$$\begin{aligned} (D_s^{BB} + R_s^B \sum_{i \neq s} R_i^{B,T} D_i^{BB} R_i^B R_s^{B,T}) x_s^{B,k+1} &= R_s^B \sum_{i=1}^{N} R_i^{B,T} (b_i^B \\ &- (L_i^{BI} x_i^{I,k+1} + L_i^{BB} x_i^{B,k}) - U_i^{BB} x_i^{B,k}). \end{aligned}$$

3 Hardware Environment

In order to make an easier interpretation of the performance tables in the next section, a short description of the used hardware environment is given. The two applications run on a workstation cluster and on a multiprocessor-shared memory architecture. The workstation cluster consists of four Hewlett Packard HP725/75MHz and of 16 SUN4 sparc2 workstations which are connected by Ethernet. The shared memory architecture is a Silicon Graphics Power Challenge with eight processors (R8000/90MHz). To give an idea of the floating point performance of the three different architectures, the theoretical peak performance and the LINPACK benchmark are listed in Table 1.

Table 1 The first column is the theoretical peak performance in Mflops and the second is the rate achieved for the decomposition of a matrix of dimension 1000.

Computer	Peak	Matrix
SUN4 Sparc10/51 (50MHz)	50	27
Hewlett Packard 725 (75MHz)	150	92
SGI Power Chall. (90MHz) 1 Proz.	360	308

The communication could be performed by interface software packages such as MPI or PVM; see [MPI94, GBD$^+$93, pvm]. On the workstation cluster the Send/Receive is realized by using UNIX sockets. The shared memory variant uses a simple copy mechanism. One of the major advantages of these interfaces is that the software can be transferred to the new platform without any modifications. Further acceleration on the multiprocessor architecture is achieved by using the machine-specific interfaces of SGI. The machine-specific interfaces directly exploit the shared memory structure and need no copy at all. For the current implementations PVM was used.

4 Applications

Point Defects

It is state-of-the-art to simulate point defects in order to obtain a consistent model for dopant diffusion; see [PJK$^+$93]. Diffusion of dopant ions (boron, phosphorus arsenic and antimony) in silicon only takes place by way of interaction with either a silicon interstitial or a silicon vacancy. In the case of inert diffusion an equilibrium concentration of point defects can be assumed. Process steps like oxidation of silicon disturb this equilibrium concentration, thus the transient evolution of point defects has to be simulated. Following Hu [Hu74], point defects are modelled by two linear diffusion equations which are coupled by a nonlinear recombination term. The generation of interstitials depends on the interface velocity between silicon and silicon oxide:

$$\begin{aligned} \frac{\partial C_I}{\partial t} &= \nabla \cdot (D_I \nabla C_I) - k(C_I C_V - C_I^* C_V^*), \quad C_I(0) = C_I^*, \qquad (4.8)\\ \frac{\partial C_V}{\partial t} &= \nabla \cdot (D_V \nabla C_V) - k(C_I C_V - C_I^* C_V^*), \quad C_V(0) = C_V^*, \qquad (4.9) \end{aligned}$$

and the boundary conditions

$$D_I \frac{\partial C_I}{\partial n} = \begin{cases} -K_I(C_I - C_I^*) + F(v_{int}) & \text{on } \Gamma \\ 0 & \text{otherwise,} \end{cases} \tag{4.10}$$

$$D_V \frac{\partial C_V}{\partial n} = \begin{cases} -K_V(C_V - C_V^*) & \text{on } \Gamma \\ 0 & \text{otherwise.} \end{cases} \tag{4.11}$$

C_I and C_V denote the concentrations of interstitials and vacancies, C_I^* and C_V^* the thermal equilibrium concentrations, D_I and D_V the diffusion coefficients, and K_I and K_V the surface recombination velocities of interstitials and vacancies. v_{int} is the velocity of the interface Γ between oxide and silicon. The computational domain is 2D.

For space discretization linear finite elements are used, time discretization is performed by the trapezoidal rule, backward difference method (TRBDF) [BCF$^+$85]. In each time step two nonlinear systems have to be solved by Newton's method. Both linear systems are solved with parallel multigrid.

The multigrid cycles are optimized with regard to the serial algorithm (V-cycles start on coarse grid); see [Kah94]. On the coarsest grid, the *serial* CGS [Son89] is applied (The *parallel* CGS leads to a communication overhead). This means that all relevant data has to be sent to a specified host before starting serial CGS there. Table 2 shows the h-independence of the multigrid algorithm. The number of multigrid cycles is nearly a constant, while the number of variables increases.

With the use of PVM, the software was transferred to the SGI Power Challenge without any changes. The difference in performance with respect to the SUN4 processor is dramatic. Thus, time measurements make only sense for a large number of variables. Table 2 shows results of the parallel multigrid algorithm for the SUN4 workstation cluster and the SGI Power Challenge.

Table 2 Speed-up of the multigrid algorithm on a SUN4 workstation cluster (first row) and on the SGI Power Challenge (second row). The point defect equations are computed with 162 to 132098 variables. The first number in the Table gives the real time for one timestep (8 Newton iterations) in seconds; the second denotes the number of multigrid cycles.

#Variables	162	578	2178	8450	33282	132098
1 domain	4/16	9/20	31/20	111/19	428/18	
					25/18	119/17
4 domains	7/16	14/20	26/20	57/20	146/18	
					7/18	30/17
16 domains	22/21	21/21	30/21	49/20	101/18	

We have also considered a parallel CGS accelerated by an additive nonoverlapping Schwarz; see also [KPW96]. The local problems associated with the additive Schwarz method are solved by the serial multigrid algorithm. For the both architecture platforms considered, the parallel multigrid works faster.

Neutron Kinetics

One major task of reactor core simulation is the calculation of neutron fluxes given by the transient multigroup neutron diffusion equations [FG81]:

$$\begin{aligned}\frac{1}{v_g}\frac{\partial \phi_g}{\partial t}(\mathbf{r},t) + \nabla \mathbf{j}_g(\mathbf{r},t) + (\Sigma_{ag} + \sum_{g'>g} \Sigma_{g'g})\phi_g(\mathbf{r},t) \qquad (4.12)\\ = \sum_{g'<g} \Sigma_{gg'}\phi_{g'}(\mathbf{r},t) + \frac{1}{\lambda}\sum_{g'=1}^{G} \Sigma_{pgg'}\phi_{g'}(\mathbf{r},t)\\ + \sum_{i=1}^{I} \chi_{dg}^{i}\lambda_i C_i(\mathbf{r},t) + S_g^{ext}(\mathbf{r},t)\ ,\end{aligned}$$

$$\mathbf{j}_g(\mathbf{r},t) + D_g \nabla \phi_g(\mathbf{r},t) = 0 \qquad (4.13)$$

$$\frac{\partial C_i}{'\partial t}(\mathbf{r},t) + \lambda_i C_i(\mathbf{r},t) = \frac{1}{\lambda}\sum_{g'=1}^{G} \Sigma_{fg'}^{i}\phi_{g'} \qquad (4.14)$$

in a 3D computational domain. ϕ_g and $\mathbf{j}_g$ are the flux and current in energy group g, v_g the neutron velocity, D_g the diffusion constant. χ_{dg}^{i} are fission spectra of delayed neutrons, S_g^{ext} is the external source, λ the eigenvalue of the reactor. $\Sigma_{ag}, \Sigma_{gg'}, \Sigma_{pgg'}$, $\Sigma_{fg'}^{i}$ are cross sections, C_i the precursor concentration of type i, λ_i the decay constant. The unknowns to be computed are $\phi_g, \mathbf{j}_g$ and C_i. At outer boundaries mixed boundary conditions (vacuum or albedo boundary conditions for a reflective medium) hold. Space discretization is done by a *nodal expansion* method (NEM) [FBW77, FG81]. The domain is subdivided into boxes, (4.12),(4.14) are integrated over the volume and (4.13) over the surfaces of each box. For time discretization implicit Euler is applied in combination with an exponential transformation technique [FG81]. The equations for precursor concentrations can be substituted into the balance equations for the fluxes. This results in the system of finest grid equations. On coarser grids a multiplicative variant of multigrid, CMR ("Coarse Mesh Rebalancing") [FBMK91], is applied. On grid levels 1 to $L-1$, the coarse mesh equations are solved by red-black Gauß-Seidel, whereas on coarsest grid level L Gaussian elimination is applied.

The basic principle of parallelization is axial domain decomposition. The axial layers of the core are distributed among the processes which solve the NEM equations on the finest grid level and the CMR equations on grid levels 2 to $L-1$.

The parallel neutron kinetics code is part of a coupled thermal hydraulics / neutron kinetics code system [KM94], the thermal hydraulics part running sequentially.

As test example an emergency power case has been chosen. The simulation is performed for three different problem sizes: the first problem with approximately 4000 boxes, the second one with 8000 and the third one with 16000 boxes on the finest grid. Table 3 shows results for a cluster of HP 725 workstations. The execution times are overall execution times including the sequential parts of the program.

In this simulation only the two finest grid levels were computed in parallel, with the number of parallel processes given in the table. The coarser grid levels were computed by a single master process.

The parallelization of the multilevel cycle leads to considerable communication between the processes. Thus much better speed-up factors are expected on a shared

Table 3 Results for the parallel neutron kinetics code on HP 725 workstations with PVM: execution time in seconds.

#Processes	Problem 1	Problem 2	Problem 3
1P	5185	8795	14533
2P	4916	6915	10568
4P	4736	6638	9476

memory computer. Table 4 (each of the first numbers) shows results for the parallel code system using again PVM as parallel platform, the neutron kinetics running on the SGI Power Challenge and the thermal hydraulics on an HP 725 workstation. The basic overhead of parallelization is the same as in the case of workstation clusters, but a Send/Receive of PVM is realized by a simple copy mechanism and the more costly communication through UNIX sockets is avoided. The speed-up factors scale only up to 6 processors; for 8 processors the communication overhead becomes dominant. Further acceleration is achieved by using machine-specific interfaces of SGI instead of PVM which need no copy at all. The results are also shown in Table 4 (the second of each set of numbers).

Table 4 Results for the parallel neutron kinetics code on the SGI Power Challenge. The first number gives the execution time in seconds with PVM as parallel platform, the second number the execution time using machine-specific interfaces.

#Processes	Problem 1	Problem 2	Problem 3
1P	1529	2788	4624
2P	909/919	1402/1370	2314/2259
4P	651/626	844/ 851	1271/1277
6P	562/506	672/ 644	945/ 946
8P	776/511	851/ 629	964/ 791

5 Conclusion

Our goal was to study the scalability of parallel domain decomposition methods in an industrial environment. Three different computer architectures were used: cluster of HP and SUN workstations, and an SGI Power Challenge, a shared memory computer. For point defect analysis, the parallel multigrid with minor modifications turned out to be a very efficient method. It is somewhat surprising, that even on a workstation cluster with its low communication performance, the method works quite satisfactorily. In neutron kinetics simulation, future work concentrates on variants of multigrid with similar convergence rates, but with higher parallel potential for a large number of processors.

REFERENCES

[BCF$^+$85] Bank R., Coughran W. M., Fichtner W., Grosse E. H., Rose D. J., and Smith R. K. (1985) Transient simulation of silicon devices and circuits. *IEEE Trans. Computer Aided Des.* 4(4): 436+.

[FBMK91] Finnemann H., Böer R., Müller R., and Kim Y. (1991) Multi-level techniques for the acceleration of nodal reactor calculations. In *Proc. Int. Top. Meeting Advances in Math. Computations and Reactor Physics.* Pittsburgh.

[FBW77] Finnemann H., Bennewitz F., and Wagner M. (1977) Interface current techniques for multidimensional reactor calculations. *Atomkernenergie* 33: 1123+.

[FG81] Finnemann H. and Grundlach W. (1981) Space-time kinetics code IQSBOX for PWR and BWR. Part I: Description of physical and thermo-hydraulic models. *Atomkernenergie-Kerntechnik* 37(3): 176+.

[GBD$^+$93] Geist A., Beguelin A., Dongarra J., Jiang W., Manchek R., and Sunderam V. (1993) *PVM 3.0 User's Guide and Reference Manual.* Oak Ridge National Laboratory, Tennessee.

[Hu74] Hu S. M. (1974) Formation of stacking faults and enhanced diffusion in the oxidation of silicon. *J.Appl.Phys.* 45: 1567+.

[Kah94] Kahlert M. (1994) Prozeßsimulation auf verteilten Rechnersystemen. Master's thesis, Mathematisches Institut der Technischen Universität München.

[KM94] Knoll A. and Müller R. (1994) Coupling RELAP5 and PANBOX2: A three-dimensional space-time kinetics application with RELAP5. In *Proceedings of Intern. Conference on New Trends in Nuclear System Thermohydraulics.* Pisa.

[KPW96] Kahlert M., Paffrath M., and Wever U. (1996) Grid partitioning versus domain decomposition: A comparison of some industrial problems on workstation clusters. *Surv. Math. Ind.* 6: 133+.

[Mey90] Meyer A. (1990) A parallel preconditioned conjugate gradient method using domain decomposition and inexact solvers on each subdomain. *Computing* 45(217+).

[MPI94] (1994) *Message Passing Interface Forum. MPI: A message-passing interface standard.*

[PJK$^+$93] Paffrath M., Jacobs W., Klein W., Rank E., Steger K., Weinert U., and Wever U. (1993) Concepts and algorithms in process simulation. *Surv. Math. Ind.* 3: 149+.

[pvm] PVM home page in the World Wide Web. World Wide Web, , http://www.epm.oml.gov/pvm/pvm_home.html.

[Son89] Sonneveld P. (1989) CGS, a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.* 10(36+).

A Shallow Water Model Distributed Using Domain Decomposition

Elías Kaplan

1 Introduction

The aim of the present work is the study of the astronomic and storm tidal currents in the Río de la Plata ($35°S, 54°W$). The hydrodynamic equations for shallow water in two dimensions, obtained by vertical integration of the Navier-Stokes equations that express the laws of mass and momentum conservation, are numerically solved employing a Leendertse scheme [LL78, Bor85].

Using the divide-and-conquer method of domain decomposition (**DD**) as the means of parallelization of the numerical solver we obtained a better response in terms of model speed and output detail thanks to flexibility in accommodating local refinement [GK92]. This allows near real-time operation of the model, coupled with transport-diffusion numerical models, for tidal prediction in ocean engineering applications.

The computation is performed in a cluster of high end workstations, communicated by message passing through a FDDI network using PVM [GBD$^+$93]. The computational domain is divided into blocks and a domain partitioner was developed to ensure a good load balancing.

2 The Shallow Water Model

Equations

The shallow water model is formed by the bidimensional unsteady equations of mass (2.1) and momentum (2.2) conservation, obtained by depth averaging Navier-Stokes [GKV$^+$92, KVR92]:

0 CeCal, Engineering School, Montevideo, Uruguay (elias@fing.edu.uy)

1 With a grant of the Conicyt-Uruguay, supported also by project ITDC'94-European Union.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

$$\frac{\partial \eta}{\partial t} + \nabla \cdot (H\vec{V}) = 0, \tag{2.1}$$

$$\frac{\partial \vec{V}}{\partial t} + (\vec{V}\cdot\nabla)\vec{V} - 2(\vec{\Phi} \times \vec{V}) + g\nabla\eta + \frac{g\vec{V}\left\|\vec{V}\right\|}{{C_h}^2 H} + \frac{\vec{\tau_s}}{\rho H} + \varepsilon\Delta\vec{V} = 0, \tag{2.2}$$

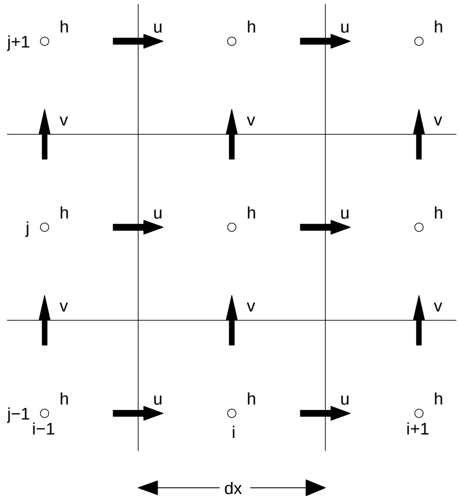
where η is the elevation of the water above the mean level and $\vec{V} = (u, v, 0)$ is the velocity in the axis coordinates (x, y, z). Moreover, H is the depth of the water, g is the gravity acceleration, $\vec{\Phi} = (0, 0, \omega)$ is the angular velocity, $-2(\vec{\Phi} \times \vec{V})$ is the Coriolis force, C_h is the Chezy coefficient, ρ is the sea water density, $\vec{\tau_s} = (\tau_{sx}, \tau_{sy}, 0)$ is the wind stress at the sea surface and ε is the coefficient for turbulent viscosity.

Advective processes (corresponding to the terms $\vec{V}\cdot\nabla\vec{V})$ are dominant in atmospheric and oceanic circulation systems governed by shallow waters equations, while diffusive effects are important only in boundary layer regions [Net92]. Any numerical model working on those equations should treat advective effects accurately.

Discretization

A finite difference scheme is employed with a staggered mesh, implicit in space and explicit with splitting in the time domain. This *Arakawa class C grid* [Ark88] has good conservative properties and is also well suited to the **DD** method with overlapped regions as employed here [AL80].

Figure 1 Distribution of the dependent variables on a two-dimensional grid.



Boundary Conditions

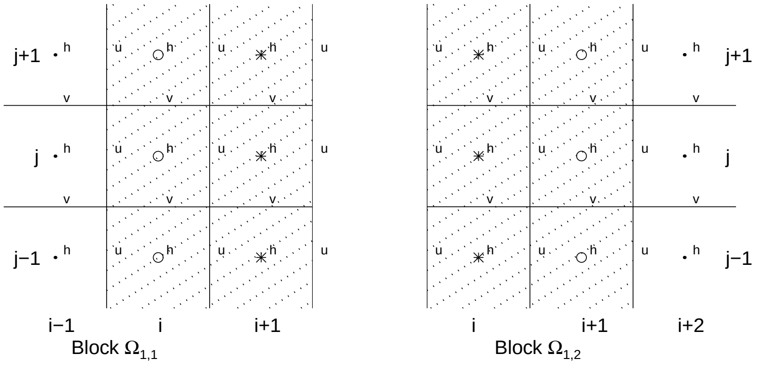
The different types of boundary conditions to impose are [AQS94]:

1. At coastlines, zero normal velocity.
2. Normal velocity is prescribed at boundaries modeling inflow from rivers.
3. Surface elevation is prescribed at south oceanic boundary, modeling a wave that enters the computational domain. Prediction of this elevation is taken from oceanic models [Sch83, Ray93].
4. Open boundary condition to simulate tidal surges that leave the computational domain [RC87]. The criterion used in this development is a variation of the Sommerfeld radiation condition (eq. 2.3):

$$\eta_t + c^x \eta_x + c^y \eta_y = 0. \tag{2.3}$$

   Here $\overrightarrow{c} = (c^x, c^y)$ and $|\overrightarrow{c}| = \sqrt{gH}$ is the wave celerity. The wave at this boundary has the same direction as the water velocity and a sense that lets the surges leave the computational domain.
5. Inter-block boundary condition between the blocks of the **DD** [DD94, Meu91]. In this artificial boundary a Dirichlet condition is imposed to the mass conservation equation (2.1) and a single iteration is performed to allow the overlapped cells to converge.

Figure 2 Inter-block boundary condition and overlapped area.

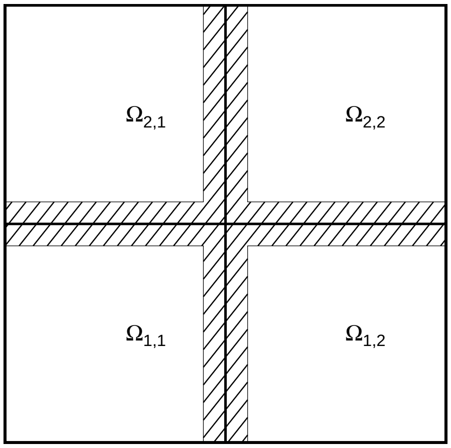


In Figure 2 the use of the overlapped region between blocks is shown. Variables marked with a $*$ are used to set boundary conditions in the the block to which they belong; variables marked with a $\circ$ are to be sent to the adjacent block for use as boundary conditions. This allows a non-blocking send between blocks, improving the performance of the model.

3 Domain Decomposition

In each block of the **DD** an **ADI** [LL75, KT92] (implicit in space, alternating directions in the horizontal and vertical successively) method is employed [LM92]. A first approximation of the data in the overlapped area between blocks, marked with a $*$ in Figure 2, is carried out using an explicit computation [CBBK94], resulting in an explicit/implicit hybrid method [DD94].

Figure 3 Example of a domain decomposition with the overlapped areas.



The explicit nature of the computation in the boundary between blocks puts a Courant (3.4) limitation in the time step, facing the less restrictive size of the purely implicit scheme [Roa72, Hir91, DD94]:

$$dt < \frac{dx}{\sqrt{g\overline{\overline{H}}}}, \tag{3.4}$$

where dx is the cell size, dt is the time step and $\sqrt{g\overline{\overline{H}}}$ represents an estimated tidal wave celerity in the area of the continental shelf inside the computational domain.

This scheme gives a speedup of the distributed model, due to the great magnitude of the number of cells to compute, over the serial model. The model is being tested in our virtual "parallel computer", a cluster of heterogeneous workstations, with a relatively high latency time and average bandwidth, but with an overall good cost/performance ratio. Better results could be achieved in machines with smaller latency time [LM92].

In the domain decomposition great care has been taken in the load balancing keeping in mind the number of cells in each block and the different performance of the workstations.

4 Results

For an improved calibration of the numerical model over the Río de la Plata, an area of the continental shelf must be included in the computational domain. The first test of the model is on a mesh of 400 by 400 km with 1 km grid size, giving 160,000 cells, and a 1 minute time step. A second mesh with 0.5 km resolution gives a total of 640,000 grid points, and a correspondingly reduced time step of 0.5 minutes.

In Figure 4 an application of the model to predict the speed field in low-tide condition is shown. In this figure block divisions are marked using dark lines. Figure 5 compares the surface elevation results of the model, while simulating the semi-diurnal tide component M_2[2], with the known astronomic wave component in Montevideo. The astronomic components at the coastal ports, used in the comparison and in boundary conditions, are obtained by harmonic analysis of several years surface elevation records [SOH, SHO].

Figure 4 Low-tide velocity field in the Río de la Plata.

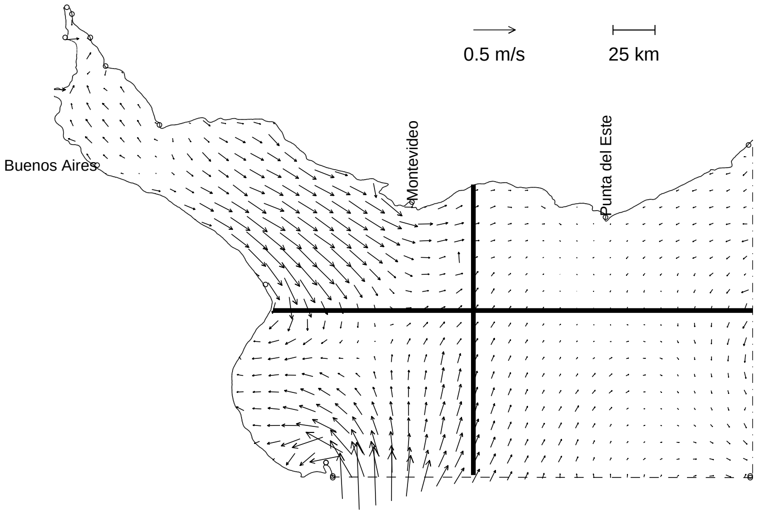


Table 1 shows the speedup of the parallelized model using 4 dedicated workstations communicating by FDDI compared to the serial model. Columns 2 and 3 pictures the Tidal Model run-time for the two meshes employed. In columns 4 and 5 of the table the run-time is measured using the Tidal Model coupled with a transport-diffusion environmental model, which overloads the workstations with a practical application, giving a better speedup in this case.

[2] This is the main astronomic tide component, it has a period of 12.4206 hours and an amplitude between 0.30 and 0.10 meters in the Argentinian and Uruguayan coasts, respectively.

[3] Tidal Model.

[4] Tidal and Transport-Diffusion Models coupled.

Figure 5 Model results and known surface elevation for M_2 semi-diurnal tide component.

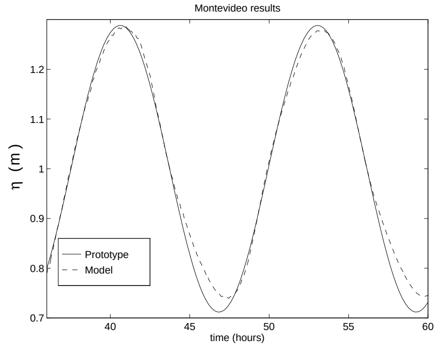


Table 1 Run time and speedup in 24 hours of simulation.

Cell size (m)	1000	500	1000	500
Number of cells	160,000	640,000	160,000	640,000
	TM^3		TM&TD^4	
Serial run time	57'	7h 30'	2h 00'	15h 00'
Parallel run time	22'	2h 20'	40'	4h 15'
Speedup	2.6	3.2	3.0	3.5

REFERENCES

[AL80] Arakawa A. and Lamb V. R. (1980) A potential enstrophy conserving scheme for the shallow water equations. *Monthly Weather Review* 109: 18–36.

[AQS94] Agoshkov V., Quarteroni A., and Saleri F. (1994) Recent developments in the numerical solution of shallow water equations I: boundary conditions. *Appl. Numer. Math.* 15(2): 175–200.

[Ark88] Arkawa A. (1988) *Physically-Based Modelling and Simulation of climate and climatic Change, Part-I*, chapter Finite-Difference methods in Climate Modelling, pages 79–168. Kluwer Academic Press.

[Bor85] Borche A. (1985) Modelo matemático de correntología do estuário do río Guaíba. Technical Report 12, Instituto de Pesquisas Hidráulicas da UFRGS, Porto Alegre, Brasil. In Portuguese.

[CBBK94] Cekirge H., Berlin J., Bernatz R., and Koch M. (1994) An appropiate algorithm in parallel computations for three-dimensional hydrodynamics. *Math. Comput. Modelling* 20(1): 65–84.

[DD94] Dawson C. and Dupont T. (1994) Explicit/implicit, conservative domain decomposition procedures for parabolic problems based in block centered finite differences. *SIAM J. Numer. Anal.* 31(4): 1045–1061.

[GBD$^+$93] Geist A., Beguelin A., Dongarra J., Mancheck R., and Sunderam V. (1993) *PVM 3.0 Users's Guide and Reference Manual.* Oak Ridge National Laboratory Report.

[GK92] Gropp W. D. and Keyes D. E. (1992) Domain decomposition methods in computational fluid dynamics. *Int. J. Numer. Meths. Fluids* 14: 147–165.

[GKV$^+$92] Guarga R., Kaplan E., Vinzon S., Rodriguez H., and Piedracueva I. (June 1992) Aplicación de un modelo de corrientes en diferencias finitas al Río de la Plata. *Revista Latinoamericana de Hidráulica* 1(4): 93–115. In Spanish, with English abstract.

[Hir91] Hirsch C. (1991) *Numerical Computation of Internal and External Flows.* John Wiley & Sons.

[KL94] Kim C. and Lee J. (1994) A three-dimensional pc-based hydrodynamic model using an ADI scheme. *Coastal Engineering* 23: 271–287.

[KVR92] Kaplan E., Vinzon S., and Rodriguez H. (1992) Aplication of a tidal currents numerical model to the Río de la Plata. In *Hydraulic Engineering Software IV, Fluid Flow Modelling*, volume 4 of *Hydrosoft*, pages 561–573. Elsevier.

[LL75] Leendertse J. and Liu S. (1975) A three-dimensional model for estuaries and coastal seas: Vol. ii, aspects of computation. Technical report, The Rand Corp., Santa Monica, California, USA.

[LL78] Liu S. and Leendertse J. (1978) Multidimensional numerical modelling of estuaries and coastal seas. Technical report, The Rand Corp., Santa Monica, Calif.

[LM92] Leca P. and Mane L. (1992) A 3-D ADI alghoritm on distributed memory multiprocesors. In Simon H. D. (ed) *Parallel Computational Fluid Dynamics, Implementation and Results*, pages 149–165.

[Meu91] Meurant G. R. (1991) A domain decomposition method for parabolic problems. *Appl. Numer. Math.* 8: 427–441.

[Net92] Neta B. (1992) Analysis of finite elements and finite differences for shallow water equations: A review. *Mathematics and computers in simulation* 34(2): 141–161.

[Ray93] Ray R. (1993) Global ocean tide models on the eve of TOPEX/POSEIDON. *IEEE Transactions on Geos. and Rem. Sensing* 31(2): 355–364.

[RC87] Roed L. and Cooper C. (1987) A study of various open boundary conditions for wind forced barotropic numerical ocean models. In *Oceanography Series*, volume 45, pages 305–335. Elsevier.

[Roa72] Roache P. (1972) *Computational Fluid Dynamics.* Hermosa Publishers.

[Sch83] Schwiderski E. (1983) Atlas of ocean tidal charts and maps: I. the semidiurnal principal lunar tide M_2. *Marine Geodesy* 6(3-4): 219–265.

[SHO] Table des mares des grands ports du monde. Serv. Hydr. et Ocean., Paris, France. In French.

[SOH] Almanaque 1987. SOHMA., Montevideo, Uruguay. In Spanish.

90

A Distributed Algorithm for 1-D Nonlinear Heat Conduction with an Unknown Point Source

C.-H. Lai

1 Introduction

The mean tool face temperature involved in intermittent cutting operations such as metal cutting and face milling, has a very important influence on the rate of tool wear and tool life. High temperatures may cause the material to fatigue or deform under face milling or other cutting operations [Sha84]. Therefore accurate simulation of temperature distributions of the work piece subject to milling or cutting is vital in order to lengthen the life time of the tool and to guarantee the quality of the cutting. In particular, real-time simulation of such temperature distributions is of industrial interests.

A major barrier in industry is that cutting temperatures are required experimentally which are then used as empirical data in suitably chosen thermal models. The measurement of physically meaningful temperatures is extremely difficult. It is particularly true for the measurement of deformation or shear zone temperatures. On the other hand, thermal models do not provide direct numerical simulation of the cutting process based on the governing differential equation [Bec85]. In order to provide a simulation software for metal cutting, a numerical algorithm is required. It is natural to assume that the application of a cutting tool at the cutting point is equivalent to the application of a source at the same point. Therefore if one can simulate the equivalent source at the cutting point, then one would be able to simulate the temperature distribution. Such approach is often referred to as an inverse problem approach. The approach is used in this paper for the modelling of a simplified cutting process. The aim of this paper is to study a domain decomposition algorithm for the simulation of temperature distributions along a work piece under cutting operations.

The layout of the paper is as follows. First, a description is given of the model for an idealised cutting problem. Second, the partitioning of the physical problem into a number of subproblems is discussed. Third, a distributed numerical algorithm is introduced. Different numerical schemes are employed in different subdomains in

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

order to solve different subproblems. Numerical tests are provided for three different types of material. An efficiency analysis is also included. Finally, some conclusions are drawn.

2 An Idealised Cutting Model

Pioneering work in remote sensor methods for the retrieval of temperature distributions can be found in [LNK67]. Recently, such methods have been developed into more mature inverse methods [CW88][Ste91][YW86] for various cutting situations. However, the use of inverse methods becomes pragmatic since analytical temperature distributions are difficult to derive. In this paper, sensors and numerical methods are combined to provide solutions to metal cutting problems.

To simplify the cutting problem, a piece of metal of infinite length and of homogeneous material property along the longitudinal direction, is considered. If the cutting tool is applied at a position along the width direction, then it is possible to assume a one-dimensional analogy of the physical problem. Therefore the domain of interest is along the width only, i.e. $x_0 < x < x_1$. Assuming the cutter is applied at $x = x_c$, then the above cutting problem can be described by the one-dimensional nonlinear unsteady parabolic heat conduction equation,

$$\frac{\partial\theta}{\partial t} = \frac{\partial}{\partial x}(k(\theta)\frac{\partial\theta}{\partial x}) + Q_c(t)\delta(x - x_c) , \tag{2.1}$$

subject to initial condition $\theta(x,0) = \Theta(x)$ and boundary conditions $\theta(x_0,t) = \Theta_0$ and $\theta(x_1,t) = \Theta_1$ where $\theta(x,t)$ is the temperature distribution, $k(\theta)$ is the conductivity of the metal, $Q_c(t)$ is the equivalent source being applied at $x = x_c$, and $\delta(x - x_c)$ is the Dirac delta function.

The continuity of the function $\frac{\partial\theta}{\partial t}$ at $x = x_c$ suggests that $\int_{x_c^-}^{x_c^+} \frac{\partial\theta}{\partial t} dx = 0$. Hence the equivalent source strength can be obtained by integrating (2.1) from $x = x_c^-$ to $x = x_c^+$ to give

$$k(\theta)\frac{\partial\theta}{\partial x}|_{x_c^+} - k(\theta)\frac{\partial\theta}{\partial x}|_{x_c^-} + Q_c(t) = 0 . \tag{2.2}$$

The above equation is used to retrieve the source strength. Hence the temperature gradients just to the left ($x = x^-$) and to the right ($x = x^+$) of the cutter must be known. A temperature sensor is attached at $x = x_s$, such that $x_s < x_c < x_1$, and let the temperature measured by means of the temperature sensor be $\theta(x_s, t) = T(t)$. However, it is not necessary to have x_s being less than x_c. The purpose of the temperature sensor is to complete the problem described in (2.1) and to allow the computation of the temperature gradients. The knowledge of the measured temperature at the sensor is then used to back-calculate average or effective measures at the cutting point. Such inverse methods avoid the basic difficulties of the direct method since remote temperatures can be measured more easily and accurately.

For computer simulation purpose, the sensor temperature is modelled by the sinusoidal function $T(t) = \alpha \sin \omega t$. Its maximum value is governed by the amplitude α and its variation with respect to time is governed by the angular frequency ω.

3 Problem Partitioning

In order to solve the inverse problem given in (2.1) with the extra condition given at $x = x_s$, the problem is partitioned into three subproblems defined in the subdomains namely, $S_1 = \{x : x_0 < x < x_s\}$, $S_2 = \{x : x_s < x < x_c\}$, and $S_3 = \{x : x_c < x < x_1\}$. The partition is basically driven by the problem at the physical level [Lai94] and the effect is to remove the unknown source term $Q_c(t)$. In other words, the differential equations in these three subdomains do not involve the Dirac delta function. Since the temperature is given at $x = x_0$ and there is a temperature sensor located at $x = x_s$, therefore Dirichlet boundary conditions are defined at the boundary of S_1. One can then solve the differential equation to obtain the derivative $\frac{\partial \theta}{\partial x}(x_s, t)$. Hence with the knowledge of the temperature $\theta(x_s, t)$ acquired by the temperature sensor at $x = x_s$, an initial value problem can be formulated in S_2. Therefore $\theta(x_c, t)$ can be obtained by solving the initial value problem. Note that k drops out because $k(u_1) = k(u_2)$ at $x = x_s$. Finally another Dirichlet problem can be formulated in S_3. Thus, we have the three subproblems as follow:

SP_1: $\frac{\partial u_1}{\partial t} = \frac{\partial}{\partial x}(k(u_1)\frac{\partial u_1}{\partial x})$ in S_1
subject to $u_1(x, 0) = \Theta(x)$, $u_1(x_0, t) = \Theta_0$, $u_1(x_s, t) = T(t)$.

SP_2: $\frac{\partial u_2}{\partial t} = \frac{\partial}{\partial x}(k(u_2)\frac{\partial u_2}{\partial x})$ in S_2
subject to $u_2(x, 0) = \Theta(x)$, $u_2(x_s, t) = T(t)$, $\frac{\partial u_2(x_s,t)}{\partial x} = \frac{\partial u_1(x_s,t)}{\partial x}$.

SP_3: $\frac{\partial u_3}{\partial t} = \frac{\partial}{\partial x}(k(u_3)\frac{\partial u_3}{\partial x})$ in S_3
subject to $u_3(x, 0) = \Theta(x)$, $u_3(x_c, t) = u_2(x_c, t)$, $u_3(x_1, t) = \Theta_1$.

The above three subproblems are well-defined [Bec85][Zwi89], and a unique solution exists for each of them. The direct sum of these subproblem solutions gives the temperature distribution of the original problem, i.e.

$$\theta(x,t) = \begin{cases} u_1(x,t), & x \in S_1 \\ u_2(x,t), & x \in S_2 \\ u_3(x,t), & x \in S_3 \end{cases} . \tag{3.3}$$

Note that the above algorithm is intrinsically sequential. However, a careful load balancing would make a distributed algorithm with minimal communication possible.

4 The Distributed Numerical Algorithm

One obvious way of distributing subproblems is to employ as many loosely coupled workstations as the number of subproblems. In the present case there should be three workstations. It is possible to treat the algorithm as a pipe line process, in which case the solution of SP_1 at a new time step will be computed first and then SP_2, etc. Therefore there is a time-lag for SP_3 compare with SP_2 and SP_1, i.e. all of the three workstations are occupied with work at the beginning of the third time step and before the last two time steps. Let W_{SPi} denotes the computational work involved in solving SP_i. The situation $W_{SP1} \neq W_{SP2} \neq W_{SP3}$ yields a distributed algorithm with computing time depending on $\max\{W_{SP1}, W_{SP2}, W_{SP3}\}$. Since the sensor is located in the neighbourhood of the cutter, therefore the subdomain S_2 is usually very small. On the other hand, $u_1(x_s,t) = T(t)$ is known, the subproblem SP_1 can be operated completely independent of SP_2 and SP_3. It is therefore possible to reduce the communication time by putting SP_2 and SP_3 into a workstation for sequential process. In such situation, the computation of the source strength does not involve inter-processor communication. To maintain load balancing in the two workstations, we require $W_{SP1} \simeq W_{SP2} + W_{SP3}$. The equality means that the two processes are synchronised. If $W_{SP1} \neq W_{SP2} + W_{SP3}$, then it is important that $\frac{\partial u_1(x_s,t)}{\partial x}$ is sent from the first processor to the second processor and is kept in the local memory of the second processor for the use in subsequent computations. For synchronised processes, such storage is not necessary. The distributed algorithm is given as:

```
Distributed Algorithm for Metal Cutting Process
Processor 1:
    for i = 1, number_of_steps
        $t := i * \triangle t$;
        Compute the solution of $SP_1$ at time $t$; Compute $\frac{\partial u_1(x_s,t)}{\partial x}$;
        Non-blocking Send $\frac{\partial u_1(x_s,t)}{\partial x}$ to Processor 2;
    end-for
Processor 2:
    for i = 1, number_of_steps
        $t := i * \triangle t$; Blocking Receive $\frac{\partial u_1(x_s,t)}{\partial x}$ from Processor 1;
        Compute the solution of $SP_2$ at time $t$;
        Compute the solution of $SP_3$ at time $t$;
        Compute $\frac{\partial u_2(x_s,t)}{\partial x}$ and $\frac{\partial u_3(x_s,t)}{\partial x}$;Retrieve $Q_c(t)$ using (2.2);
    end-for
```

The meaning of non-blocking send in the above algorithm is that computation in the sending processor resumes as soon as the message is safely on its way to the receiving processor. The meaning of blocking receive in the above algorithm is that the receiving processor has to wait until the correct message from the sending processor has arrived.

Numerical Schemes

A first order forward difference approximation of the temporal derivative and a second order central difference approximation of the spatial derivatives are used in the subproblems SP_1 and SP_3. An explicit scheme is resulted from the difference approximation. Dropping the subscripts used in denoting the subdomains, the explicit scheme for SP_1 and SP_3 can be written as

$$u_i^{(n+1)} = rb_i^{(n)}u_{i-1}^{(n)} + (1 - r(a_i^{(n)} + b_i^{(n)}))u_i^{(n)} + ra_i^{(n)}u_{i+1}^{(n)} , \tag{4.4}$$

where i denotes the i-th grid point, $r = \frac{\triangle t}{(\triangle x)^2}$, $a_i^{(n)} = \frac{k_{i+1}^{(n)}+k_i^{(n)}}{2}$, $b_i^{(n)} = \frac{k_i^{(n)}+k_{i-1}^{(n)}}{2}$, (n) denotes the time-step, $\triangle t$ is the step size along the temporal axis and $\triangle x$ is the mesh size along the spatial axis x.

The subproblem SP_2 becomes a second order initial value problem along the spatial dimension when the first order backward difference approximation, denoted as m, of

Table 1 Conductivity parameters.

Material	a	b	c	d
A	0.5	-1.1	1.0	-1.0
B	0.8	-0.5	0.01	-0.01
C	1.0	-0.3	0.0001	-0.0001

the temporal derivative at $x = x_s$ is substituted into $\frac{\partial u_2}{\partial t}$ of SP_2. A one-step modified Euler integration scheme is applied to solve the initial value problem and is written as in the following pair of calculations,

$$\begin{pmatrix} u \\ v \end{pmatrix}^* = \begin{pmatrix} u \\ v \end{pmatrix} + \triangle x \underline{f} \, , \quad \begin{pmatrix} u \\ v \end{pmatrix}^{\text{new}} = \begin{pmatrix} u \\ v \end{pmatrix} + \frac{\triangle x}{2} \{ \underline{f} + \underline{f}^* \} \, , \tag{4.5}$$

where $v = \frac{\partial u}{\partial x}$, $\underline{f} = \underline{f}\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} v \\ \frac{m}{k} - \frac{k'}{k} v^2 \end{pmatrix}$ and $\underline{f}^* = \underline{f}\begin{pmatrix} u \\ v \end{pmatrix}^*$. A second order accurate solution may be obtained for each of the three subproblems. Therefore it is expected to have a second order accurate global solution for the inverse problem (2.1). The effect of global truncation errors for SP_2 is minimised because of the small size of the subdomain which usually consists of only a few modified Euler steps. Since the numerical schemes in SP_1 and SP_3 are explicit, therefore the CFL condition, $\triangle t \leq \frac{\triangle x^2}{2K}$ where $K = \max\{k_i^{(n)}\}$, must be satisfied.

A sequential Fortran program has been written to perform the above tasks. PVM (Parallel Virtual Machine) [Gei94] is used to provide distributive directives in order that the tasks can be distributed onto a network of Sun workstations. For the present studies, only two Sun workstations are required.

Numerical Tests

A number of tests was performed by taking the conductivity as $k(u) = a + bu + cu^2 + du^3$, where a, b, c, and d are parameters used to describe the material property of a piece of metal. Table 1 shows three sets of different conductivity parameters used in the subsequent tests. The nonlinearity of the conductivity decreases as $|c|$ and $|d|$ decreases. In particular the last set of conductivity parameters represents almost a constant conductivity. For the conductivity as shown in Table 1, α and ω are chosen to be 0.4 and 2π respectively. The boundary points are located at $x = x_0$ and x_1. The initial value $\Theta(x)$ and the boundary values Θ_0 and Θ_1 are chosen to be zeros. The locations of the sensor and the cutter, i.e. x_s and x_c, are chosen to be 0.4 and 0.5 respectively. Numerical results are provided for two mesh sizes $\triangle x = \frac{1}{20}$ and $\triangle x = \frac{1}{40}$ and the corresponding $\triangle t$'s are chosen to be 0.001 and 0.0002. The resulting numbers of modified Euler steps in SP_2 are two and four respectively. Temperature distributions at $t = 1.1$ seconds are shown in Figure 1. The results show that $\triangle x$ has little effect on the temperature distribution. Source strength variations with respect to t for different mesh sizes are also similar.

Efficiency Analysis

In order to study the parallel computational work, the number of elementary operations per grid point per time-step is needed. The number of elementary operations involved in the numerical schemes are listed in Table 2. In this analysis, it is assumed that the workstations are loosely coupled in a local network such as Ethernet. It is easy to work out the number of operations for the explicit scheme as 43. Let t_s be the CPU time required to perform 43 floating point operations, then the CPU time required to march one time-step in SP_1 and SP_3 are n_1t_s and n_3t_s respectively where n_1 and n_3 are the number of grid points in SP_1 and SP_3. By counting the operations involved in the modified Euler's method, the CPU time required to march one time-step forward in SP_2 is $\frac{56}{43}n_2t_s$. Therefore the following parallel computational time for n time steps is estimated,

$$t_p = n_1t_s + \frac{56}{43}n_2t_s + n_3t_s + (n-1)\max\{n_1t_s, \frac{56}{43}n_2t_s + n_3t_s\} + t_c$$

where t_c is the average communication time between any two workstations. The speed-up can then be estimated as

$$S = (n_1 + \frac{56}{43}n_2 + n_3)t_sn/t_p \tag{4.6}$$

It is natural to ignore t_c during evening or weekend runs, and as $n \to \infty$, the ideal speed-up is obtained as

$$S = \frac{n_1 + \frac{56}{43}n_2 + n_3}{\max\{n_1, \frac{56}{43}n_2 + n_3\}} \tag{4.7}$$

The relation

$$n_1 = \frac{56}{43}n_2 + n_3 \tag{4.8}$$

is used to check the load balancing of the distributed algorithm. For the case $n_1 > \frac{56}{43}n_2 + n_3$, $S = 1 + \frac{56}{43}\frac{n_2}{n_1} + \frac{n_3}{n_1}$, for the case $n_1 = \frac{56}{43}n_2 + n_3$, $S = 2$, and for the case $n_1 < \frac{56}{43}n_2 + n_3$, $S = (\frac{56}{43}\frac{n_2}{n_1} + \frac{n_3}{n_1})^{-1} + 1$. Therefore, the speed-up ratio satisfies $1 < S_p \leq 2$, for any positive integer n_1. If the problem size approaches the limits $\frac{n_2}{n_1} \to 0$ and $\frac{n_3}{n_1} \to 1$, then the ideal speed-up approaches 2. Figures 2 and 3 confirm the above result by plotting CPU times against various problem sizes in both sequential and distributed runs. The results shown in these figures were run on SUN SPARC5 workstations connected locally by Ethernet.

The analysis above shows that the use of multi-step methods for the solutions of the initial value problem in SP_2 is not recommended because the communication time will increase. However, if SP_1 is a small subdomain which requires SP_2 to be solved with SP_1 in a processor in order to achieve load balance, then multi-step methods can be employed which do not increase the communication time.

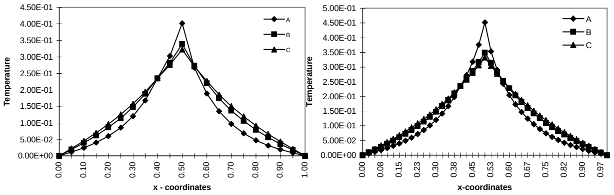
5 Conclusions

The use of the distributed algorithm for the retrieval of heat source at the cutter and the calculation of the temperature distribution is presented. PVM is used to examine

Table 2 Operations count.

Elementary Operations									
$k(u)$	$k'(u)$	r	m	$\frac{k'}{k}$	a_i	b_i	$\underline{f}$	$\begin{pmatrix} u^* \\ v^* \end{pmatrix}$	$\begin{pmatrix} u \\ v \end{pmatrix}$
9	7	2	2	17	20	11	23	27	29

Figure 1 Temperature distributions for $h = \frac{1}{20}$ and $h = \frac{1}{40}$ at $t = 1.1$s.



the efficiency of the algorithm in a distributed environment. Numerical results show that the algorithm is scalable. Fast and parallel numerical schemes may then be used within individual subproblems to reduce the overall computing time.

REFERENCES

[Bec85] Beck J. (1985) *Inverse Heat Conduction.* John Wiley & Son Inc., New York.

[CW88] Chow J. and Wright P. (1988) On-line estimation of tool/chip interface temperature for a turning operation. *ASME Journal of Engineering for Industry* 110: 56–64.

[Gei94] Geist A. (1994) *PVM - a User Guide and Tutorial for Networked Parallel Computing.* MIT Press, London.

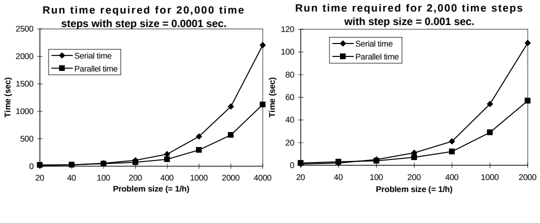
[Lai94] Lai C.-H. (1994) Diakoptics, domain decomposition and parallel computing. *The Computer Journal* 37: 840–846.

[LNK67] Lipman M., Nevis B., and Kane G. (1967) A remote sensor method for determining average thermal properties heated by moving heat sources. *ASME Journal of Engineering for Industry* 89: 333–338.

[Sha84] Shaw M. (1984) *Metal Cutting Principles.* Oxford University Press, London.

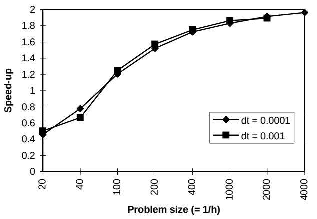
[Ste91] Stephenson D. (1991) An inverse method for investigating deformation zone temperatures in metal cutting. *ASME Journal of Engineering for Industry* 113:

Figure 2


Run time required for 20,000 time steps with step size = 0.0001 sec.

Run time required for 2,000 time steps with step size = 0.001 sec.

Figure 3 Speedup ratio.



129–136.

[YW86] Yen D. and Wright P. (1986) A remote temperature sensing technique for estimating the cutting interface temperature distribution. *ASME Journal of Engineering for Industry* 108: 252–263.

[Zwi89] Zwillinger D. (1989) *Handbook of Differential Equations.* Academic Press Inc., San Diego.

Domain Decomposition of an Atmospheric Transport–Chemistry Model

Andreas Müller

1 Introduction

The simulation of temporal and spatial distribution of reactive constituents of the atmosphere relies on a set of coupled nonlinear partial differential equations in space, time and the considered species. The numerical approach to the solution of these equations of balance is the discretization of space and time. A numerical solution of this system has especially to consider that the temporal scales relevant to the meteorological processes differ largely from those relevant to chemical processes. Therefore the system of balance equations is split into two systems - a transport- and a chemistry system. In every time–step the model first solves for each species the transport component. The transport–system uses extensively the concept of operator–splitting - the transport–equation is split into an advection- and a diffusion–equation ([Str68]). Each of these three–dimensional equations is decomposed into a series of it's one–dimensional components. The one–dimensional advection–equations are solved by the FCT–method, the diffusion–equations are solved by a modified second–order Lax-Wendroff–method ([BB76],[vL74]). After this at each grid point the rate of change of each species concentration resulting from the chemical kinetics - expressed by a set of coupled, nonlinear ordinary differential equations - is solved by the chemistry solver. Because of the stiffness of the system a semi–implicit method is used ([GJM82]).

At the Institut für Meteorologie und Klimaforschung of the Research Center Karlsruhe and of the University Karlsruhe for regional scales a sophisticated numerical model system consisting of the meteorological model KAMM [AF73], the transport model DRAIS [TD88] and the chemical reaction model RADM [CBI$^+$87] has been developed since years. KAMM models the velocity field necessary for DRAIS whereas RADM evaluates the chemical interaction equations. The presentation at hand concentrates on the model parts DRAIS and RADM because these parts need nearly the whole computation time of the model.

Typical grids consist of 50000–100000 grid points on which the physical processes are

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

simulated with time-steps up to 20 seconds. In contrast the chemistry simulation works with time-steps between 0.1 and 5 seconds. A one-day simulation with 26 species takes about 12 hours computing time by using the vector computer VP400. An overview of explicit meteorological applications and visualization of numerical results is given in [Fie93].

The method of domain decomposition is used to parallelize the solution process of the model system. It will be shown that the use of domain decomposition is a good possibility to shorten the response time on parallel computers strongly. Another advantage of this method is that on parallel computing systems it can be engaged to simulate larger problems because of the data can be maintained on the local memories of the computing system.

2 Governing Equations

Neglecting molecular diffusion the spatial and temporal distribution of the concentration field $c_s(\boldsymbol{r}, t)$, $s = 1, \ldots, n$ of a set of n species is given by the system of balance equations:

$$\frac{\partial c_s}{\partial t} + \boldsymbol{\nabla} \cdot (c_s \boldsymbol{v}) = S_{c_s}, \qquad s = 1, \ldots, n, \tag{2.1}$$

where $\boldsymbol{v}$ is the wind velocity and S_{c_s} are source and sink terms due to for example emissions, chemical reactions or deposition at the ground.

Due to the typical different time scales relevant to the various processes under consideration each equation is split into an homogeneous and an inhomogeneous part in view of the numerical solution of the system. The homogeneous part of the balance equations refers to the meteorological transport and the inhomogeneous part to the chemical reactions. The latter is accepted to be independent of spatial derivations. Thus the solution of (2.1) will be the replaced by solving (2.2) and (2.3) successively

$$\frac{\partial c_s}{\partial t} + \boldsymbol{\nabla} \cdot (c_s \boldsymbol{v}) = 0 \qquad s = 1, \ldots, n \tag{2.2}$$

$$\frac{\partial c_s}{\partial t} = S_{c_s} = f_s(c_1, \ldots, c_n, t) \qquad s = 1, \ldots, n. \tag{2.3}$$

Applying these equations to the turbulent system of the atmosphere all variables ϕ are decomposed into a mean $\overline{\phi}$ and a fluctuating part ϕ' (Reynolds–decomposition). In chemistry the fluctuating parts are neglected. The turbulence is closed with a first order parameterization ([MY74]). So under the assumption of shallow convection ([DF69]) we can write the equations of conservation (2.2) as

$$\frac{\partial \overline{c_s}}{\partial t} + \boldsymbol{\nabla} \cdot (\overline{c}_s \overline{\boldsymbol{v}}) - \overline{c}_s (\boldsymbol{\nabla} \cdot \overline{\boldsymbol{v}}) = \boldsymbol{\nabla} \cdot (\mathcal{K} \cdot \boldsymbol{\nabla} \overline{c_s}), \tag{2.4}$$

with $\mathcal{K}$ the tensor of diffusion. Because of the generally not flat bottom of the model terrain following systems of coordinates are used. By a transformation of the irregular z-coordinate the model volume is transformed into a cube ([GCS75]). Because of that numerical methods for structured grids can be employed.

3 Numerical Solution

Figure 1 shows the sequence of calculation. In order to solve the meteorological transport equation (2.4) this equation is decomposed into a three–dimensional advection and into a three–dimensional diffusion equation ([Str68]). Both are solved in a transport–step. After this the rate of change of the species concentrations is calculated by the chemistry–simulator. The numerical scheme at the n–th step we can express by operators:

$$\boldsymbol{c}_{n+1} = \mathcal{C} \circ (\mathcal{A} + \mathcal{D})\boldsymbol{c}_n$$
$$\mathcal{D} = \sum \mathcal{D}^i$$

with the operators $\mathcal{A}, \mathcal{C}$ and $\mathcal{D}$ referring to advection, diffusion and chemical composition. The index i characterizes the spatial direction.

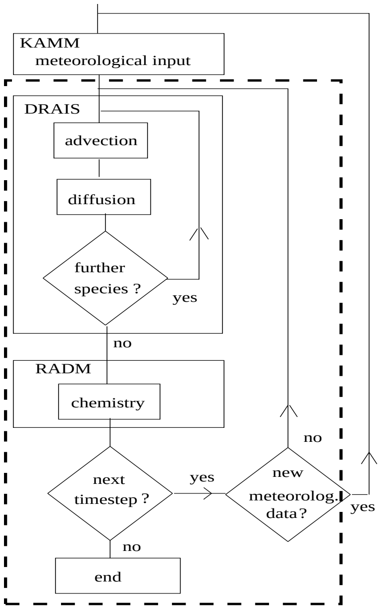


Figure 1 Flowchart of the simulation

The transport–simulator splits the three–dimensional advection- and diffusion equations in the one–dimensional compounds ([Str68]). The time–integration of these equations uses forward-Euler discretization. The one–dimensional equations of advection are solved by the FCT–method (flux–corrected–transport). The diffusion equations are solved by a modified second order Lax–Wendroff difference–method ([BB76],[vL74]). The transport equations have to be solved for each species. Their solution is independent of each other.

To model the chemistry a stiff system of nonlinear ordinary equations

$$\frac{\partial c_s}{\partial t} = f_s(c_1, \dots, c_n, t) \quad , s = 1, \dots, n. \tag{3.5}$$

has to be solved. On the assumption of a production rate a_s of a species s (independent of its own concentration) and a loss rate $b_s \cdot c_s$ we can write equation 3.5 in the following form:

$$\frac{dc_s}{dt} = f_s(c_1, \dots, c_n, t) = a_s - b_s \cdot c_s \quad , s = 1, \dots, n. \tag{3.6}$$

Because of the coupling of different species by chemical reactions the functions a and b are in general not constant but dependent on the concentrations of other species. The algorithm is a predictor-corrector integration scheme with internal self-adaptive time–steps. [GJM82].

4 Domain Decomposition

The topological regular grid is decomposed into sub–grids by hyper-planes parallel to planes formed by two coordinate axes.

Now we want to discuss the use of a different number of coordinates for domain decomposition (dimension) with regard to the parallelization of the problem. So the aim of this work is to investigate the benefit the domain decomposition for acceleration without changing anything in the physical and the numerical processes of modeling and solution.

For the first we consider a transport–step of the constituents. In a transport–step three one–dimensional advection and three one–dimensional diffusion equations are solved by explicit schemes. Meanwhile a transport–step the solution of advection and diffusion equations in a sub-domain doesn't need any data of an other sub-domain [Mül96].

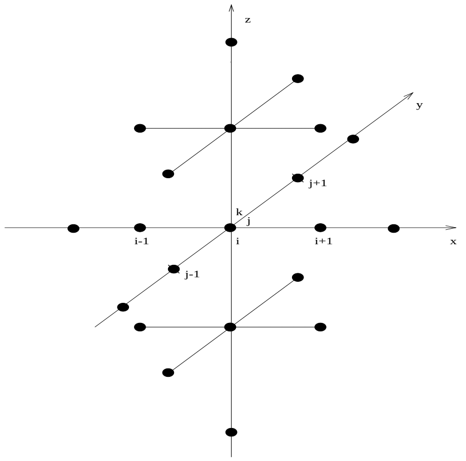


Figure 2 Structure of data dependence in a grid point G_{ijk} for a transport–step

The union of all six difference stars at a grid point G_{ijk} (figure 2) gives the grid points from which the calculation scheme for the transport at a fixed interior grid point needs data from the foregoing chemistry step. Between different solution steps of a transport–step their is no communication necessary. Because of the equal number of arithmetical operations for the calculation of the transport in each grid point a homogeneous distribution of the grid points on different processors leads also to a good balance of the numerical load between the processors. By domain decomposition we distribute the domain (one–dimensional) in the following way.

Let the grid be decomposed by $n-1, n \geq 2$ hyper-planes

$$H_j = \{(x,y,z) \in G; y = c_j\},\ j \in J,$$

in n disjunctive sub–grids G_j

$$\begin{aligned} G &= \bigcup_{j \in J} G_j, \\ G_j \cap G_{j'} &= \emptyset \ \text{ for } \ j, j' \in J, \\ J &= \{\iota; \iota = 0, \dots, n-1\}. \end{aligned}$$

Each of the n processors gets for simulation the sub–grid G_j defined by

$$G_j := [1, IX] \times [y_r(j), y_s(j)] \times [1, IZ]$$

IX, IY and IZ are the number of grid points in $x-, y-$ and z-direction. With

$$B = \lfloor IY/IYP \rfloor$$

and

$$R = IY \pmod{IYP}$$

we can define the $y-$index of the boundary grid points of the sub-domains by (l:left,r:right):

$$\begin{aligned}
y_l(0) &:= 1 \\
y_r(0) &:= \begin{cases} B+1 & \text{for} \quad R > 0 \\ B & \text{for} \quad R = 0 \end{cases} \\
y_l(j) &:= y_r(j-1)+1 \\
y_r(j) &:= \begin{cases} y_l(j)+B+1 & \text{for} \quad j < R \\ y_l(j)+B & \text{for} \quad j \geq R. \end{cases}
\end{aligned}$$

The different numerical load at the physical boundary grid points is not relevant for the considered problem size.

After the simulation of the change of the concentrations caused by advection and diffusion the change of the concentration caused by chemistry is calculated. Using the same domain decomposition for the chemistry–module there is no communication from the transport step necessary because the simulation of the chemical interaction runs at each grid point independent of other grid points. Before the beginning of the next meteorological step communication has to take place.

In figure 3 the used computation time for a chemistry–step is shown for different processors summed over all vertical grid point layers of the model. Because of the higher gradients of the concentrations of most species the need of computation time in the layers near the bottom is higher. (The vertical layers are countered from the top.) The reason therefore is a smaller internal time-step for grid points with spatial high gradients and so for a fixed time interval between transport and chemistry locally differing number of iterations of the implicit method.

So we use the common domain decomposition only in one or two horizontal components. The decomposition of the vertical components by horizontal sections on different processors is not suited because of the inhomogeneous distribution of work between the different horizontal layers which is caused by the implicit chemical solver.

The distribution of the calculation of sub-domains to different processors leads to logical boundary grid points. The calculation at these grid points needs data from ghost points which exist in the memory of other processors.

Figure 4 shows the structure of the communication of the two–dimensional decomposition. Data on grid points (ghost points) lying in the dark areas are sent by corresponding processors. Processors dealing with domains without physical boundaries have to communicate with four other processors, others communicate with a corresponding number of processors.

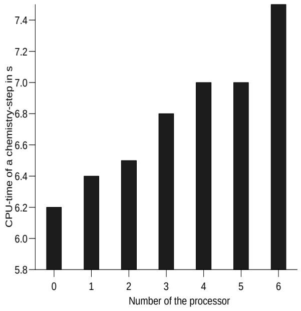


Figure 3 Distribution of CPU-time of a chemistry–step between seven processors

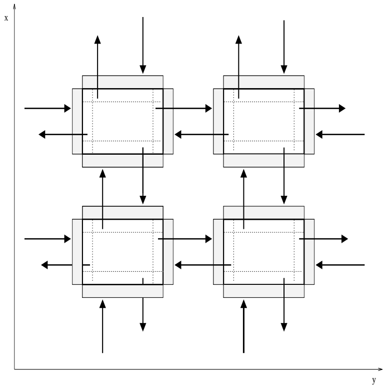


Figure 4 Symbolic representation of the communication of processors dealing with sub-domains without physical boundaries for a two–dimensional decomposition

5 Results

The implementation of the described parallelization has been performed on the MIMD–computer Paragon XP/S using explicit message passing. The grid size is 49 × 53 × 25. In each grid point 26 different species have been considered.

The measurements are given in table 1. The two–dimensional parallelization is based on a dividing of the x- and the y-axes. Let IXP and IYP be the numbers of parts IX and IY are decomposed into. Results are given in table 2.

The main loss of efficiency documented in table 1 and 2 is caused by the different calculation efforts at different processors. The main reasons for this are properties of the division of entire numbers (a nonzero remainder leads to a load imbalance between different processors) and the imbalance in the chemistry-model which is a consequence of the implicitness of the solver. So the refinement of the decomposition leads to increasing differences in calculation time meanwhile the chemistry–step between processors dealing with different sub-domains [Mül96].

Up to 70 processors an efficiency greater than 50% has been measured so that this strategy is well suited for parallelizing the model. Grids with a linear increasing number of grid points in the horizontal layers can be expected to be calculated in the same time if an appropriate linear increase of the number of the involved processors is given [Mül96].

Table 1 CPU-time and efficiency for one–dimensional decomposition

processors	computation time	efficiency
1	90.1	1.00
2	46.6	0.97
3	31.9	0.94
5	20.0	0.90
7	14.2	0.91
9	11.7	0.86
13	8.7	0.80
17	7.6	0.70
25	5.8	0.62

Table 2 CPU-time and efficiency for two–dimensional decomposition

$IXP \times IYP$	processors	computation time	efficiency
(2,4)	8	12.8	0.88
(2,12)	24	5.2	0.72
(4,6)	24	5.4	0.70
(2,25)	50	3.4	0.53
(4,17)	68	2.6	0.51
(6,13)	78	2.7	0.43
(4,25)	100	2.2	0.41
(8,17)	136	2.0	0.33

Acknowledgement

This work was been while the author was at the Institute für Meteorologie und Klimaforschung of the Research Center Karlsruhe and of the University of Karlsruhe. The implementation and the measurements of the parallel program were done at the Paragon XP/S of the Research Center Jülich.

REFERENCES

[AF73] Adrian G. and Fiedler F. (1973) Simulation of unstationary wind and temperature fields over complex terrain and comparison with observations. *Beitr. Phys. Atmosph.* 64: 27–48.

[BB76] Book D. L. and Boris J. P. (1976) Flux corrected transport I, SHASTA, A fluid transport algorithm that works. *J. of Comp. Phys.* 22: 517–533.

[CBI$^+$87] Chang J. S., Brost R. A., Isaksen I. S. A., S. Madronich P., Middleton, Stockwell W. R., and Walcek C. J. (1987) A three-dimensional eulerian acid deposition model: Physical concepts and formulation. *J. Geophys. Res.* 92: 14681–14700.

[DF69] Dutton J. A. and Fichtl G. H. (1969) Approximate equations of motions for gases and liquids. *J. Atm. Sci.* 26.

[Fie93] Fiedler F. (1993) Development of meteorological computer models. *Interdisciplinary Science Reviews* 18.

[GCS75] Gal-Chen T. and Sommerville R. C. J. (1975) On the use of a coordinate transformation for the solution of the navier-stokes equations. *J. of Comp. Phys.* 17: 209–228.

[GJM82] G. J. McRae W. R. Goodin J. H. S. (1982) Numerical solution of the atmospheric diffusion equation for chemically reacting flows. *J. of Comp. Phys.* 45: 1–42.

[Mül96] Müller A. (1996) Parallelisierung numerischer Verfahren zur Beschreibung von Ausbreitungs- und chemischen Umwandlungsprozessen in der atmosphärischen Grenzschicht. *Wissenschaftliche Berichte des Instituts für Meteorologie und Klimaforschung der Universität Karlsruhe* 18.

[MY74] Mellor G. L. and Yamada T. (1974) A hierarchy of turbulence closure models for planetary boundary layers. *J. Atm. Sci.* 31: 1791–1806.

[Str68] Strang G. (1968) On the construction and comparison of difference schemes. *SIAM, J. Numer. Anal.* 5: 506–517.

[TD88] Tangermann-Dlugi G. (1988) Numerische Simulationen atmosphärischer Grenzschichtströmungen über langestreckten mesoskaligen Hügelketten bei neutraler thermischer Schichtung. *Wissenschaftliche Berichte des Instituts für Meteorologie und Klimaforschung der Universität Karlsruhe* 2.

[vL74] van Leer B. (1974) Towards the ultimate conservative difference scheme . ii. monotonicity and conservation combined in a second–order scheme. *J. of Comp. Phys.* 14: 361–370.

92

Domain Decomposition Methods for Three-Dimensional Thermoelastic Problems on Parallel Computers

Ralf Quatember and Wolfgang L. Wendland

1 Introduction

For the computation of thermoelastic and thermoelastic-plastic stress distributions in engine parts, boundary element methods are used, for instance by the Mercedes-Benz company. These methods are preferred to FE methods because of the decisively faster mesh generation (presently three instead of twelve months). Moreover, substructuring arising from the geometrical data and different material constants on subdomains can nowadays be realized efficiently on parallel computers. In substructuring, coupling interfaces have to be introduced in the interior of the original domain.

For the computation of complex structures, serial high performance computers as the CRAY C 94 are no longer able to handle the large systems of equations and data. As a result, only rather simple machine parts can currently be simulated with the traditional software.

In this work, we present a domain decomposition algorithm which is suitable also for rather complex problems. A numerical comparison between a sequential, a (data-) parallel, and a domain decomposition boundary element program is presented.

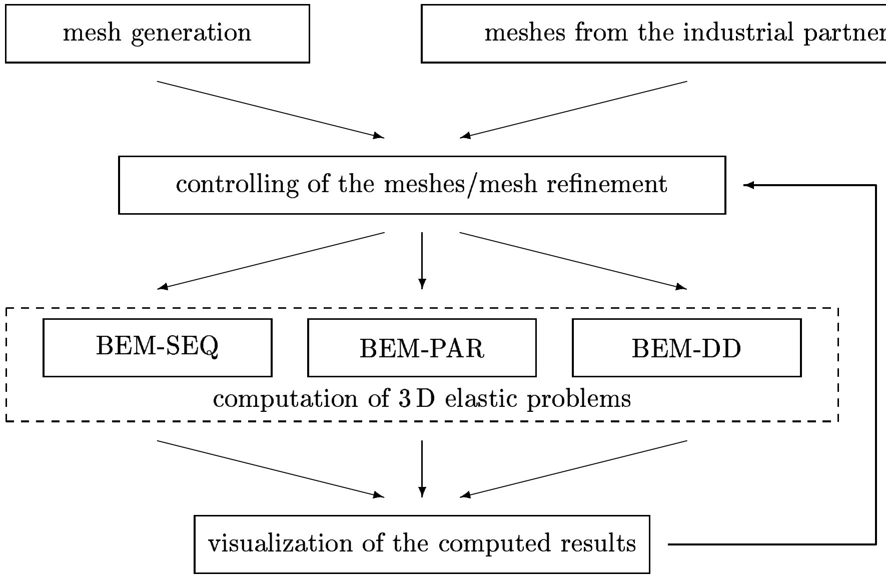
For the practical work, a pure "number crunching" program is not sufficient. Since we have to solve problems in $\mathbb{R}^3$, the visualization of the input data and the computed results is an important part of our work as well. Moreover, it is necessary to generate test meshes for the three-dimensional problems.

Figure 1 shows a typical flow-chart for our project, which is divided into the following tasks:

1. Generation of the mesh and input data (tractions and displacements) or the adaption of meshes and input data given by our partner from industry.
2. Controlling of the mesh and mesh refinement if necessary.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Figure 1 A typical flow-chart of our project


mesh generation
meshes from the industrial partner
controlling of the meshes/mesh refinement
BEM-SEQ
BEM-PAR
BEM-DD
computation of 3 D elastic problems
visualization of the computed results

3. Solution of the three-dimensional elasticity problem. Here we can use one of our three boundary element programs:
   (a) BEM-SEQ: sequential boundary element program,
   (b) BEM-PAR: (data-)parallel boundary element program. This means that the system of equations is generated distributedly and solved in parallel on the family of processors, and
   (c) BEM-DD: boundary element program for the solution of the problem by a domain decomposition algorithm on multiple processors.
4. Visualization of the computed results.

In [QSW97] we present a program which handles the visualization and the mesh generation of surface meshes in $\mathbb{R}^3$. This program manages triangles as well as quadrangles on the boundary surface, which are described by linear or quadratic form functions. In section 2 we describe our domain decomposition algorithm and in section 3 we present some numerical results.

2 Domain Decomposition Formulation

Let us consider a three-dimensional thermo-elastic body in the domain $\Omega \subset \mathbb{R}^3$ with given displacements g on the boundary part Γ_D and given boundary stresses h on the remaining boundary part Γ_N.

We further assume that a temperature field $\theta(x)$ is given in Ω. Then the volume forces $\tilde{f}$ can be split into a thermo-elastic (Duhamel-Neumann material law [Kup79]) and an elastic part:

$$\tilde{f}_i = f_i - \frac{\partial}{\partial x_i}\left[\Big(2\mu(x) + 3\lambda(x)\Big)\,\alpha(x)\,\theta(x)\right] \text{ for } i = 1, \dots, 3. \tag{2.1}$$

Here α is the given coefficient of linear heat expansion. For given volume forces we can write the static equilibrium equations as

$$\sigma_{ij,j}(u,x) + \tilde{f}_i(x) = 0 \text{ for } i = 1, \dots, 3 \text{ and } x \in \Omega, \tag{2.2}$$

where the stress tensor σ_{ij} is related to the strain tensor e_{ij} by Hooke's law

$$\sigma_{ij}(u,x) = \delta_{ij}\lambda(x)\sum_{k=1}^{3} e_{kk}(u,x) + 2\mu(x)\,e_{ij}(u,x). \tag{2.3}$$

λ and μ are the well-known Lamé constants. The linear strain-displacement relations are given by

$$e_{ij}(x) = \frac{1}{2}\Big(u_{i,j}(x) + u_{j,i}(x)\Big), \tag{2.4}$$

where $u_{\cdot,j}$ denotes the partial derivative with respect to x_j. Assuming a given non-overlapping domain decomposition

$$\overline{\Omega} = \bigcup_{i=1}^{p} \overline{\Omega}_i \text{ with } \Omega_i \cap \Omega_j = \emptyset \text{ for } i \neq j, \quad \Gamma_i = \partial\Omega_i, \quad \Gamma_{ij} = \Gamma_i \cap \Gamma_j, \tag{2.5}$$

we call $\Gamma_S = \bigcup_{i=1}^{p} \Gamma_i$ the skeleton. The Lamé constants in (2.3) are assumed to be piecewise constant on each of the subdomains:

$$\mu(x) = \mu_i, \qquad \lambda(x) = \lambda_i \text{ for } x \in \Omega_i \text{ and } i = 1, \dots, p. \tag{2.6}$$

For each subdomain Ω_i, the Kelvin fundamental solution is defined by

$$U^i_{kl}(x,y) = \frac{\lambda_i + \mu_i}{8\pi\mu_i(\lambda_i + 2\mu_i)}\left[\frac{\lambda_i + 3\mu_i}{\lambda_i + \mu_i}\,\frac{1}{|x-y|}\,\delta_{kl} + \frac{(x_k - y_k)(x_l - y_l)}{|x-y|^3}\right]. \tag{2.7}$$

Let $\big((T^i_{kl}(\cdot,\cdot))\big)$ be the corresponding boundary stress of the field of the fundamental solution. Then the solution of the differential equation (2.2) is given by the Somigliana representation formula for $x \in \Omega_i$ in each of the subdomains Ω_i:

$$c_{kl}u^i_k(x) = \int\limits_{\Gamma_i} U^*_{kl}(x,y)\,t^i_k(y)\,ds_y - \int\limits_{\Gamma_i} T^*_{kl}(x,y)\,u^i_k(y)\,ds_y + \int\limits_{\Omega_i} U^*_{kl}(x,y)\,\tilde{f}^i_k(y)\,ds_y. \tag{2.8}$$

The solution u^i satisfy the boundary conditions on the individual parts of the exterior boundary,

$$u^i(x) = g \text{ for } x \in \Gamma_D \text{ and } t^i(x) = h \text{ for } x \in \Gamma_N \tag{2.9}$$

and the coupling conditions on the skeleton,

$$t^i(x) + t^j(x) = 0 \text{ and } u^i(x) - u^j(x) = 0 \text{ for all } x \in \Gamma_{ij}. \tag{2.10}$$

For the computation of the unknown Cauchy data (u_i, t_i) we use the integral equation resulting from (2.8):

$$\left(V_i t^i\right)(x) = \left(\frac{1}{2}I + K_i\right)u^i(x) + \left(N_i \tilde{f^i}\right)(x) \text{ for } x \in \Gamma_i. \tag{2.11}$$

V_i denotes the single layer potential, K_i denotes the double layer potential and N_i denotes the Newton potential. Introducing the Steklov-Poincaré operator by the equation

$$S_i := V_i^{-1}\left(\frac{1}{2}I + K_i\right) \tag{2.12}$$

we obtain from (2.11) the Dirichlet-Neumann mapping

$$t^i = S_i u^i + V_i^{-1} N_i \tilde{f} = S_i u^i + f^i. \tag{2.13}$$

With this mapping, the equivalent variational formulation for the solution of the mixed boundary value problem (2.2) can be written as:

Find $u \in H^{1/2}(\Gamma_S)$ with $u|_{\Gamma_D} = g$ and $u^i = u|_{\Gamma_i}$ such that

$$\sum_{i=1}^{p} \int\limits_{\Gamma_i} S_i u^i(x) v^i(x) ds_x = \int\limits_{\Gamma_N} h(x) v(x) ds_x - \sum_{i=1}^{p} \int\limits_{\Gamma_i} f^i v^i(x) ds_x \tag{2.14}$$

holds for all $v \in H^{1/2}(\Gamma_S)$ with $v|_{\Gamma_D} = 0$.

Let Γ_H be a triangulation of the boundary $\Gamma = \partial\Omega$ with maximal mesh size H and

$$u_H(x) = \sum_{k=1}^{N_u} u_k\, {}'^{\nu_u}_k(x) \in V_H \subset H^{1/2}(\Gamma_S), \tag{2.15}$$

a finite representation of the displacements with respect to a B-spline basis of degree ν_u. On the subspace V_H, the variational problem (2.14) leads to

$$\sum_{i=1}^{p} \int\limits_{\Gamma_i} S_i^h u_H^i(x) v^i(x) ds_x = f(v_H), \tag{2.16}$$

where S_i^h are approximations of the local Steklov-Poincaré operators S^i on an appropriately chosen fine grid boundary element discretization W_h^i on Γ_i,

$$S_i^h\, v_H := t_h^i \in W_h^i \subset H^{-1/2}(\Gamma_i). \tag{2.17}$$

t_h^i can be found by the solution of the local finite-dimensional variational problem

$$\left\langle V_i\, t_h^i, \tau_h \right\rangle_{L^2(\Gamma_i)} = \left\langle \left(\frac{1}{2}I + K_i\right) u_H^i(x), \tau_h \right\rangle_{L^2(\Gamma_i)} \quad \text{for all } \tau_h \in \widetilde{W}_h \tag{2.18}$$

With appropriate $\widetilde{W}_h$, this formulation is valid for Galerkin methods as well as for collocation methods for the computation of the approximations of the local Steklov-Poincaré operators S_i^h.

The unique solvability of the variational problem (2.14) results from the W_h^i-ellipticity and boundedness of the local Steklov-Poincaré operators S_i^h [HW92].

For a sufficient refinement h with $h < cH$, the positive definiteness of the approximate operators S_i^h follows for the Galerkin scheme by the Strang lemma and therefore implies for $H \to 0$ the convergence of the approximate solutions of (2.16) to the solution of (2.14) and (2.2), respectively [HSW95].

Equation (2.16) together with the constraints 2.18) leads to the following algorithm:

1. Choose for the Dirichlet data along the coupling and Neumann boundaries a start solution $u^0_{|\Gamma_S}$ with $u_{|\Gamma_D} = g$.
2. Solve pure Dirichlet problems for the realization of the local Steklov-Poincaré operators and compute the corresponding local Neumann data t_i^k.
3. Check the compatibility conditions along the coupling and Neumann boundaries. If a given accuracy is reached, stop the algorithm.
4. Otherwise correct the Dirichlet data $u^{k+1}_{|\Gamma_C}$ with a preconditioned iterative method and continue with step 2.

This algorithm leads to a linear system of equations of the form

$$M_h^T V_h^{-1} \left(\frac{1}{2} \tilde{M}_h + K_h \right) \underline{u} = \underline{f} \tag{2.19}$$

with a positive definite stiffness matrix and an unsymmetric perturbation. For the solution of (2.19) one can use the minimal correction method [QSW96, SN89] as well as some generalized methods of conjugate gradients, like BiCGStab [vdV92] or GMRES [SS86].

For the preconditioning of these methods we use a hierarchical splitting of the unknown function $u \in H^{1/2}(\Gamma_S)$ according to (2.15), (2.18). Let ω be the q-dimensional set of all coarse grid nodes of our domain decomposition (2.5). Then there exists a unique splitting

$$u(x) = \sum_{i=1}^{q} u_j {}'^q_j(x) + \tilde{u}(x) \tag{2.20}$$

with

$$u_j = u(x_j), \qquad \tilde{u}(x_j) = 0 \text{ for } x_j \in \omega \tag{2.21}$$

and with the "harmonic" basis functions ${}'^q_j(\cdot)$ solving the differential equation (2.2) in the subdomains Ω_i. This leads to the coupled variational formulation:

Find the pair $(u_H, \tilde{u})$ assuming the given Dirichlet data on the boundary Γ_D such that the coarse grid system

$$\int\limits_{\Omega} \sigma_{ij}(u_H)\, e_{ij}(v_H) dx + \sum_{i=1}^{p} \left\langle S_i^h \tilde{u}_{|\Gamma_i}, v^H_{|\Gamma_i} \right\rangle_{\Gamma_i} = f_1(v^H) \tag{2.22}$$

and the fine grid compatibility conditions

$$\sum_{i=1}^{p} \left\langle S_i^h (u^H + \tilde{u})_{|\Gamma_i}, \tilde{v}_{|\Gamma_i} \right\rangle_{\Gamma_i} = f_2(\tilde{v}) \tag{2.23}$$

hold for all test functions $(v^H, \tilde{v})$ *which are zero on* Γ_D.

The coarse grid system is nothing more than a finite-element formulation with respect to the given domain decomposition with harmonic basis functions. Due to this property, the global stiffness matrix can be computed with the local Steklov-Poincaré operators. Moreover, the solution on this coarse grid can be interpreted as a mapping $u^H = R\tilde{u}$ of the fine grid function $\tilde{u}$ onto coarse grid functions.

Inserting this mapping into the compatibility conditions, we have to find a fine grid solution which satisfies the modified compatibility conditions

$$\sum_{i=1}^{p} \left\langle S_i (\tilde{u} + R\tilde{u})_{|\Gamma_i}, \tilde{v}_{|\Gamma_i} \right\rangle_{\Gamma_i} = f(\tilde{v}) \tag{2.24}$$

for all test functions $\tilde{v}$.

The solution process includes only one additional step to solve the coarse grid system for any given fine grid solution. Namely, replace in the algorithm at the beginning of this section the function $u_{|\Gamma_C}$ by $\tilde{u}_{|\Gamma_C}$ and insert between steps 1 and 2 the additional procedure

1.5 Solve for the current fine grid solution $\tilde{u}^k_{|\Gamma_C}$ the coarse grid system and compute the momentary complete iterate $u^k_{|\Gamma_C}$.

For the solution of the variational problem (2.24) along the coupling interfaces Γ_{ij} we need appropriate preconditioning. A reasonable choice is the Neumann-Neumann preconditioner [LT94]

$$M^{-1} = \sum_{i=1}^{p} S_i^{-1}, \tag{2.25}$$

which uses the inverse Steklov-Poincaré operators S_i^{-1}, that can be computed by the solution of local mixed Dirichlet-Neumann or pure Neumann problems. The solution of pure Neumann problems needs some more care because one has to incorporate equilibrium equations. Here we use a modified Neumann series [HW72] in an extended algorithm including a coarse grid solver.

3 Numerical Examples

As a practical example we analyze and compute a three-dimensional crank shaft from our industrial partner Mercedes-Benz, discretisized with 872 boundary elements and 438 boundary points. The computation was performed on one hand with our sequential and (data-)parallel programs, and on the other hand with the domain decomposition algorithm presented here, in particular with a decomposition into two subdomains.

Presently we are developing a program which can treat more than two subdomains and a better memory management such that finer discretizations can be handled, too.

All our computations were performed on a SUN SparcStation 20 with a 90 MHz HyperSparc processor and 128 MB main memory; and also on a Parsytec PowerXplorer with eight MPC601 processors and 32 MB main memory for each processor. The presented computation times are pure processor times or processor plus communication times. For the determination of the efficiencies we compared the computation times (including the communication times) on 2, 4 and 8 processors with the computation time (without communication) on one processor.

Figure 2 Example for a practical computation: crank shaft (Mercedes-Benz)

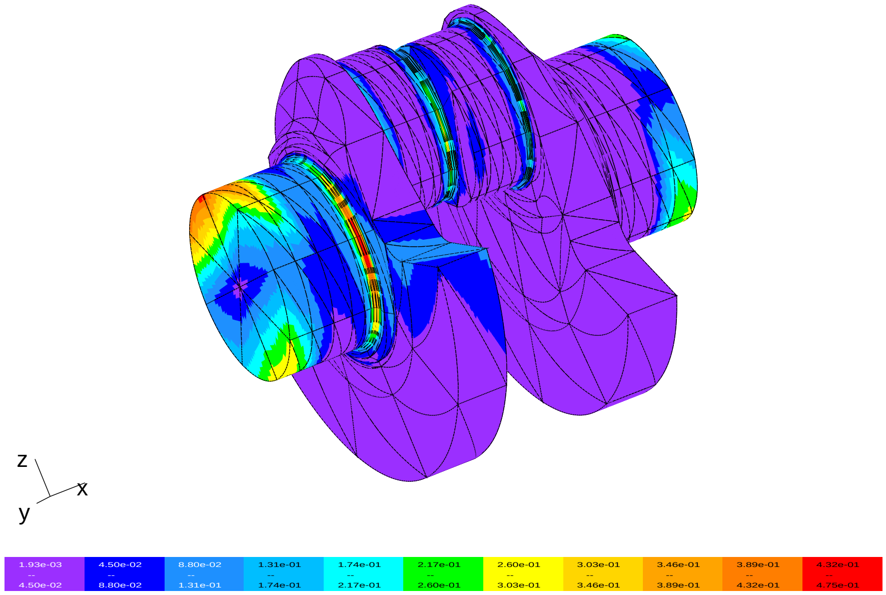


Table 1 Computational times for the numerical computation of the crank shaft discretisized with 872 elements and 438 points (2616 unknowns)

	PowerXplorer					SUN
	BEM-PAR				BEM-DD	BEM-SEQ
Proc:	1	2	4	8	2	1
time (sec):	249.6	143.0	76.9	44.6	139.3	198.5
efficiency:	100%	87.3%	87.3%	70.0%	89.6%	—

Figure 2 shows the result of these computations. In case of the iterative solution a stopping tolerance of 10^{-4} was chosen. Table 1 shows the computation times needed.

For finer discretizations and subdivisions in more than two subdomains we believe that the comparison between the different algorithms will result even more in favour of the domain decomposition method.

Numerical results for the significantly simpler example of the Laplacian in two dimensions and a subdivision into 16 subdomains obtained by O. Steinbach in [Ste94] show the efficiency of the presented algorithm and we expect similar results for the three-dimensional case.

REFERENCES

[HSW95] Hsiao G. C., Schnack E., and Wendland W. L. (1995) A hybrid coupled finite-boundary element method. Technical Report 95–11, Universität Stuttgart.

[HW72] Haack W. and Wendland W. L. (1972) *Lectures on Partial and Pfaffian Differential Equations.* Pregamon Press, Oxford.

[HW92] Hsiao G. C. and Wendland W. L. (1992) Domain decomposition via boundary element methods. In Alder H. *et al.* (eds) *Numerical Methods in Engineering and Applied Sciences*, pages 198–207. CIMNE, Barcelona.

[Kup79] Kupradze V. D. (1979) *Three-Dimensional Problems of the Mathematical Theory of Elasticity and Thermoelasticity.* North-Holland, Amsterdam.

[LT94] Le Tallec P. (1994) Domain decomposition methods in computational mechanics. *Comput. Mech. Advances* 1: 121–220.

[QSW96] Quatember R., Steinbach O., and Wendland W. L. (1996) Domain decomposition based solvers for industrial stress analysis with boundary elements. In Neunzert H. (ed) *Progress in Industrial Mathematics at ECMI 94.* John Wiley & Sons and B. G. Teubner, Stuttgart.

[QSW97] Quatember R., Steinbach O., and Wendland W. L. (1997) Entwicklung vorkonditionierter Iterationsverfahren für Randelementmethoden der Thermoelastizität auf Parallelrechnern. In Hoffmann K.-H., Jäger W., Lohmann T., and Schunck H. (eds) *Mathematik — Schlüsseltechnologie für die Zukunft*, pages 191–201. Springer, Heidelberg.

[SN89] Samarskij A. A. and Nikolaev E. S. (1989) *Numerical Methods for Grid Equations.* Birkhäuser-Verlag, Basel.

[SS86] Saad Y. and Schultz M. H. (1986) GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* 7: 856–869.

[Ste94] Steinbach O. (1994) Boundary elements in domain decomposition methods. *Contemp. Math.* 180: 343–348.

[vdV92] van der Vorst H. A. (1992) BI-CGSTAB: A fast and smoothly converging variant of BI-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* 13: 631–644.

93

A Domain Decomposition Solver for Three-Dimensional Steady Free Surface Flows

Einar M. Rønquist

1 Introduction

The prediction of the free surface shape is an important aspect of many materials processing applications. It poses great challenges in terms of the design of efficient solution algorithms for the governing equations, in particular, for three-dimensional problems. The governing equations are highly coupled, and there is a great need for fast, robust and memory efficient methods. Even though recent improvements in solution strategies have made three-dimensional simulations tractable [HP91, CMBBC91, EPW92, RE96], there is still room for significant improvement.

In [Røn96], a domain decomposition method was proposed for the solution of the steady, incompressible Navier-Stokes equations, but in the context of imposing velocity boundary conditions. This paper represents an extension of the method in [Røn96] to general stress boundary conditions. In particular, we consider steady free surface problems in which surface tension plays an important role. The segregated solution approach used in [HR94] for two-dimensional problems is here extended to the three-dimensional case. The update of the geometry is decoupled from the solution of the steady Navier-Stokes equations; this paper emphasizes the latter step.

In terms of providing insight into why the method in [Røn96] works as well as it does, we refer to [Cas96]. In [Cas96] other solution approaches are also proposed for the steady Navier-Stokes equations. However, no numerical results exist yet for these alternative approaches.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 Governing Equations

We consider here steady, incompressible, Newtonian, viscous fluid flow in a three-dimensional domain Ω, as governed by the incompressible Navier-Stokes equations

$$\begin{aligned} \rho u_j u_{i,j} &= \tau_{ij,j} + f_i \qquad \text{in } \Omega\,, && (2.1)\\ u_{i,i} &= 0 \qquad \text{in } \Omega\,. && (2.2) \end{aligned}$$

Here, the stress tensor τ_{ij} can be expressed as

$$\tau_{ij} = -p\delta_{ij} + \mu(u_{i,j} + u_{j,i})\,. \tag{2.3}$$

In (2.1)-(2.3), u_i is the velocity, p is the pressure (relative to zero ambient), f_i is the body force, ρ is the density, μ is the viscosity and δ_{ij} is the Kronecker delta symbol. We use indicial notation, with subscript indicial comma denoting differentiation, e.g., $u_{i,j} = \partial u_i/\partial x_j$. Summation over repeated indices is assumed.

The domain boundary $\partial\Omega$ is decomposed as $\partial\Omega = \partial\Omega_v \cup \partial\Omega_\sigma \cup \partial\Omega_s \cup \partial\Omega_o$, with velocity boundary conditions imposed on $\partial\Omega_v$, surface tension traction boundary conditions imposed on $\partial\Omega_\sigma$, symmetry conditions imposed on $\partial\Omega_s$ and outflow boundary conditions imposed on $\partial\Omega_o$. The appropriate boundary conditions can then be expressed as

$$\begin{aligned} u_i &= \overline{u}_i && \text{on } \partial\Omega_v && (2.4)\\ n_i\tau_{ij}n_j &= \sigma\kappa && \text{on } \partial\Omega_\sigma && (2.5)\\ t_i\tau_{ij}n_j &= t_i\sigma_{,i} && \text{on } \partial\Omega_\sigma && (2.6)\\ t_i\tau_{ij}n_j &= 0 && \text{on } \partial\Omega_s && (2.7)\\ n_i u_i &= 0 && \text{on } \partial\Omega_s && (2.8)\\ n_i\tau_{ij}n_j &= -\overline{p} && \text{on } \partial\Omega_o && (2.9)\\ t_i u_i &= 0 && \text{on } \partial\Omega_o && (2.10) \end{aligned}$$

Here, $\overline{u}_i$ is the prescribed velocity on $\partial\Omega_v$ (this specification also includes the specification of no-slip conditions), n_i is the outward unit normal on the surface, t_i is any tangent vector on the surface, σ is the surface tension, κ is twice the mean curvature, and $\overline{p}$ is the ambient pressure. Note that the $t_i\,\sigma_{,i}$ in (2.6) must be interpreted as the surface gradient of the surface tension on $\partial\Omega_\sigma$. From (2.7)-(2.10) we also note that the symmetry conditions represent zero normal velocity and zero tangential stress, while the outflow conditions represent zero tangential velocity and normal stress equal to minus the ambient pressure.

Finally, the kinematic condition at steady state is

$$u_i n_i = 0 \qquad \text{on } \partial\Omega_\sigma\,. \tag{2.11}$$

3 Variational Formulation

From (2.4)-(2.10) we note that all the three velocity components are specified on $\partial\Omega_v$, two components are specified on $\partial\Omega_o$, one component is specified on $\partial\Omega_s$ and no component is specified on $\partial\Omega_\sigma$. We first define the space $H^1_D(\Omega)$ to be the usual H^1 space, but where the (scalar) members of the space must be compatible with both the non-homogeneous and homogeneous *Dirichlet* (vector) velocity boundary conditions on the domain boundary $\partial\Omega$. In an analogous fashion, we define the space $H^1_0(\Omega)$ to be similar to the space $H^1_D(\Omega)$ except that *homogeneous* Dirichlet conditions are imposed wherever Dirichlet velocity boundary conditions are specified in $H^1_D(\Omega)$.

The equivalent weak form of (2.1)-(2.10) can then be expressed as: Find $u_i \in H^1_D(\Omega)$ and $p \in L^2(\Omega)$ such that $\forall v_i \in H^1_0(\Omega)$,

$$\int_\Omega \{\, \rho v_i u_j u_{i,j} + v_{i,j}[-p\delta_{ij} + \mu(u_{i,j} + u_{j,i})] - v_i f_i \,\} d\Omega - I_\sigma(v_i) = 0 \;, \tag{3.12}$$

$$\forall q \in L^2(\Omega)\,, \qquad \int_\Omega q\, u_{i,i}\, d\Omega = 0 \qquad , \tag{3.13}$$

where

$$I_\sigma(v_i) = \int_{\partial\Omega_\sigma} v_i\, \tau_{ij} n_j \, dS = \int_{\partial\Omega_\sigma} v_i\, \overline{g}_i \, dS \;\; . \tag{3.14}$$

The surface integral given in (3.14) corresponds to the natural imposition of stress boundary conditions along the free surface. In [HP91] a very elegant form is derived for the imposed surface traction $\overline{g}_i$, where only surface-intrinsic co-ordinates are needed, and where a single form automatically generates both the normal and tangential boundary conditions required for viscous analysis.

In this study, our variational form is based on the results derived in [HP91]. Even though the numerical results in [HP91] put a restriction on the free surface to be either closed or periodic, the variational results derived in [HP91] are also appropriate for the more general case where the free surface intersects an outflow boundary or a symmetry boundary. For example, the variational form automatically provides a term which allows for the imposition of contact angles in three dimensions; this term is needed in this study in order to impose a 90° angle along the line where the free surface intersects the symmetry boundary or the outflow boundary.

For free surface problem, the geometry along the free surface is an unknown in addition to the velocity and pressure. The variational form derived in [HP91] only requires the geometry space to include C^0-surfaces, even though the curvature term in (2.5) (strong form) suggests that C^1-surfaces may be required. The lowering of the continuity requirement for the geometry description is here in line with finite element analysis for second-order partial differential equations.

The kinematic condition given in (2.11) can be used directly in order to update the free surface shape. Here, however, we shall use an approach similar to the approach studied in [HR94] for two-dimensional problems. In particular, we shall consider an extension to three dimensions for the case when surface tension plays an important role; this is described in the next section.

4 Discretization and Solution Strategy

Our discrete equations are generated based upon the weak form. Following a spectral element discretization procedure [MP89], the domain is broken up into K hexahedral elements. Within each element, and along each (local) spatial direction, the geometry and the velocity are approximated as N^{th} order polynomials, while the pressure is approximated as a polynomial of degree $N-2$ [MPR92].

Following a similar procedure as described in [Røn96] (non-staggered grid), we arrive at a set of discrete equations which can be expressed in matrix form as

$$\underline{\mathbf{A}}\,\underline{\mathbf{u}} + \underline{\mathbf{C}}(\underline{\mathbf{u}})\,\underline{\mathbf{u}} - \underline{\mathbf{D}}^T \underline{p} = \underline{\mathbf{f}} \ , \tag{4.15}$$

$$\underline{\mathbf{D}}\,\underline{\mathbf{u}} = \underline{0} \ . \tag{4.16}$$

Here, $\underline{\mathbf{A}}$ is the discrete (coupled) viscous operator, $\underline{\mathbf{C}}(\underline{\mathbf{u}})$ is the discrete, nonlinear, nonsymmetric advection operator. $\underline{\mathbf{D}}$ is the discrete divergence operator, and its transpose $\underline{\mathbf{D}}^T$ is the discrete gradient operator. The vector $\underline{\mathbf{u}}$ contains the nodal velocity values, $\underline{p}$ represents the nodal pressure values, and the components of $\underline{\mathbf{f}}$ are the nodal forces.

We now give a brief summary of the entire solution procedure:

1. Similar to the approach used in [HR94], we first solve the discrete Navier-Stokes equations imposing the *kinematic* condition along the free surface (zero normal component) instead of the original *free surface* boundary conditions.
2. Next, we evaluate the residual in the discrete momentum equations imposing the *original* free surface boundary conditions, and restrict this residual to the free surface. We remark that, since we assume that surface tension plays an important role (large Weber number), we can *interpret* the normal component of this free surface residual as being due to an incorrect local curvature used in (3.14).
3. The surface term (3.14), represented in the form derived in [HP91], is linearized with respect to the normal surface displacement. From this linearization, we create a system of equations for the normal free surface displacement, with a right hand side equal to the residual in the momentum equations, restricted to the free surface, and restricted to the normal component.
4. We now solve the system for the normal free surface displacement.
5. The free surface displacement is extended smoothly to the interior of the computational domain by solving the three-dimensional, discrete elasticity equations, imposing the free surface displacement as Dirichlet data.
6. The geometry is now updated, taking into account the computed deformation in the entire domain.
7. Steps 1-6 are repeated until the residual in the momentum equations and the free surface displacement are below prescribed tolerances.

5 A Domain Decomposition Navier-Stokes Solver

In the following, we shall focus our attention on Step 1 in the algorithm described in the previous section. We start by first expressing the original, nonsymmetric, nonlinear, discrete steady Navier-Stokes system (4.15)-(4.16) in the compact form

$$\underline{\mathbf{F}}\,\underline{\mathbf{x}} = \underline{\mathbf{g}} \ , \tag{5.17}$$

where

$$\begin{aligned}
\underline{\mathbf{F}} &= \begin{pmatrix} (\underline{\mathbf{A}} + \underline{\mathbf{C}}(\mathbf{u})) & -\underline{\mathbf{D}}^T \\ \underline{\mathbf{D}} & \underline{\mathbf{0}} \end{pmatrix} , \\
\underline{\mathbf{x}} &= [\underline{\mathbf{u}}, \underline{p}]^T , \\
\underline{\mathbf{g}} &= [\underline{\mathbf{g}}_u, \underline{g}_p]^T \equiv [\underline{\mathbf{f}}, \underline{0}]^T .
\end{aligned}$$

As usual we linearize the system (5.17) and perform a Newton iteration. For each iteration we have to solve a system of the form

$$\underline{\mathbf{N}}\,\delta\underline{\mathbf{x}}^n = \underline{\mathbf{g}} - \underline{\mathbf{F}}\,\underline{\mathbf{x}}^{n-1} \tag{5.18}$$

where $\underline{\mathbf{N}}$ represents the linearized Navier-Stokes operator, $\underline{\mathbf{x}}^n$ is the solution after n Newton iterations, and $\delta\underline{\mathbf{x}}^n = \underline{\mathbf{x}}^n - \underline{\mathbf{x}}^{n-1}$.

The iterative substructuring algorithm we now present is for the system (5.18). Our method can therefore be described as a Newton-Krylov method. We proceed by directly considering the preconditioned, linearized, steady Navier-Stokes system

$$\underline{\mathbf{M}}^{-1}\underline{\mathbf{N}}\,\delta\underline{\mathbf{x}}^n = \underline{\mathbf{M}}^{-1}(\underline{\mathbf{g}} - \underline{\mathbf{F}}\,\underline{\mathbf{x}}^{n-1}) , \tag{5.19}$$

where $\underline{\mathbf{M}}^{-1}$ represents the Navier-Stokes preconditioner.

Following the approach given in [Røn96], the Navier-Stokes preconditioner can be expressed in the additive form

$$\underline{\mathbf{M}}^{-1} = \underline{\mathbf{M}}_0^{-1} + (\mathbf{I} + \underline{\mathbf{M}}_\Gamma^{-1}\underline{\mathbf{G}})\,[\sum_{k=1}^{K} \underline{\mathbf{M}}_k^{-1}\,] + \underline{\mathbf{M}}_\Gamma^{-1} \tag{5.20}$$

where

$$\underline{\mathbf{G}} = \begin{pmatrix} \underline{\mathbf{0}} & +\underline{\mathbf{D}}^T \\ \underline{\mathbf{0}} & \underline{\mathbf{0}} \end{pmatrix} . \tag{5.21}$$

We now briefly comment on the individual components in this preconditioner. Most of the details are given in [Røn96] in the context of the velocity formulation of the Navier-Stokes equations. We shall here emphasize the extension of this preconditioner to solving problems with general boundary conditions, for which the stress formulation of the Navier-Stokes equations is required. Our specific application will be the study of three-dimensional free surface flow. The original work in [Røn96] was inspired by progress in the understanding and application of domain decomposition methods over the past years [DW90, SBG96].

The first term in the preconditioner, $\underline{\mathbf{M}}_0^{-1}$, involves the restriction of the velocity-pressure residual to a coarse grid. We consider here a coarse grid based upon quadratic Q_2/P_1 finite elements induced by the original spectral element decomposition. Note that, in this study, each N^{th}-order spectral element represents an individual subdomain. The quadratic finite element grid is used in order to construct a coarse discretization of the original, linearized Navier-Stokes operator. We also add streamline diffusion on the coarse grid if the grid Peclet number is sufficiently large. The coarse residual represents the right-hand-side for this coarse system, which is solved using a direct, banded solver. Note that the coarse, linearized Navier-Stokes system here couples all the velocity components, even for the linear Stokes case. The coarse system is also compatible with all the original boundary conditions, in which zero, one, two or three velocity components are specified on the various surface segments. After the coarse solution has been computed, the solution is extended to the original spectral element mesh.

The term $\underline{\mathbf{M}}_k^{-1}$ in the preconditioner represents the solution of a local Stokes problem defined in subdomain $\Omega_k, k = 1, ..., K$, with homogeneous velocity boundary conditions prescribed on the subdomain boundary $\partial\Omega_k$. We now make several remarks regarding the local, discrete Stokes operator. First, the stress formulation is replaced by the velocity formulation. This is a valid replacement due to the homogeneous velocity boundary conditions. Thus, the coupled viscous operator is replaced by the (vector) Laplacian operator. Second, the scalar, spectral, elemental Laplacian operator is replaced by a spectrally equivalent finite element operator [DM90]; in three dimensions, this operator represents linear tetrahedral elements based on the tensor-product Gauss-Lobatto Legendre points. The local, scalar, discrete finite element Laplacian operator is explicitly constructed and inverted using a banded, direct solver. In order to proceed, this operator is then used to explicitly construct and invert the local, Uzawa pressure operator (the local pressure is defined to have a zero average). Hence, only back substitution is needed during each preconditioning step, one to find the local pressure, and one for each velocity component. For more details, see [Røn96].

The term $\underline{\mathbf{M}}_\Gamma^{-1}$ represents the inversion of the diagonal of the viscous operator for all the velocity degrees-of-freedom on the subdomain interfaces, Γ. Note that this operator appears in two places in the preconditioner (5.20). This is due to the fact that there are two contributions to the right-hand-side for this interface system. One is the initial residual in the momentum equations, restricted to the subdomain interfaces. The second is the result of operating with the gradient operator $\underline{\mathbf{D}}^T$ on the pressure found from solving all the local Stokes problems, and then restricting the result to the subdomain interfaces; this second contribution gives rise to the gradient operator $\underline{\mathbf{G}}$ in (5.20). Finally, the prolongation operator associated with this interface system represents the identity operator for the velocity degrees-of-freedom along Γ, and the zero operator for the remaining degrees-of-freedom in the domain.

In summary, our Navier-Stokes preconditioner is based upon a *hierarchy* of discrete, spatial operators, starting with a linearized Navier-Stokes operator for the coarse, global problem, a steady Stokes operator for each individual, local problem, and finally, an elliptic (Poisson type) operator for the interface problem.

Finally, we remark that all the individual components in the preconditioner (5.20) are here updated after each geometry update. This is perhaps a conservative approach, and future work will include an investigation into finding a more optimal updating strategy for free surface problems.

6 Numerical Results

We now present numerical results for the extrusion of a three-dimensional die through an extruder with a square cross-section. Symmetry conditions allows us to only consider a quarter of the die. The boundary conditions consist of: (i) the specification of an inlet velocity; (ii) no-slip velocity conditions along the extruder walls; (iii) free surface boundary conditions along the surface of the die that is exposed to ambient pressure; (iv) outflow conditions to limit the computational domain; and (v) two orthogonal symmetry planes resulting from the fact that we only consider a quarter of the original, square die.

The computational domain is broken up into $K = 33$ spectral elements, each of order $N = 5$. The Reynolds number is $Re = \rho UL/\mu = 20$ and the Weber number is $We = \sigma/(\rho LU^2) = 2$. Here, the characteristic length, L, is based upon the thickness of the die, and the characteristic velocity, U, is based upon the inlet velocity.

Figure 1 shows the initial surface mesh along the (fixed) extruder walls (right half of the domain), along the free surface (left half of the domain), and along one of the symmetry planes (bottom surface). The flow goes from right to left. The fixed, square extruder exit is represented by the square cross-section in the middle of the domain.

Figure 2 shows the final surface mesh along the fixed extruder walls (identical to the initial mesh), along one of the symmetry planes (the view is identical to the initial mesh), along the free surface, and along the outflow boundary. As the die exits from the extruder (in the middle of the domain), and a relaxation of the stresses occur, we see that the die swells away from the corners of the square extruder exit, while the die contracts near the corners of the extruder (only a single corner is present in this computational model due to the two orthogonal symmetry planes). The die shape also undergoes a dramatic change in shape from having a square cross section at the extruder exit to having a circular cross section at the outflow boundary.

We now comment on the outflow conditions used during this simulation. A priori, the pressure at the outflow is not known. However, we made the *assumption* that the final die shape at the outflow would, in fact, have a circular shape. In this case, the pressure can readily be calculated from knowing the radius of the cross-section and the surface tension. For each update of the geometry, this model was used in order to update the pressure at the outflow boundary. Note that only a *single node* along the intersection of the free surface and the outflow boundary was used for the purpose of estimating the radius. At convergence, we see that our assumption was correct, and hence, our strategy is valid for this particular case.

In terms of the performance of the steady Navier-Stokes algorithm, our experience so far suggests that the behavior is quite similar to the one reported in [Røn96]. For simple, three-dimensional model problems, the introduction of the more general stress formulation as well as more general stress boundary conditions, do not seem to significantly change the earlier performance results.

Acknowledgement

The author would like to thank NASA Goddard Space Flight Center for the financial support of most of this work (contract NAS5-38075). The author would also like to thank Dr. Lee W. Ho for many helpful discussions and suggestions during the course of this work. Finally, the author would like to thank Prof. Mario A. Casarin and Prof. Olof B. Widlund for their comments on an earlier version of the manuscript.

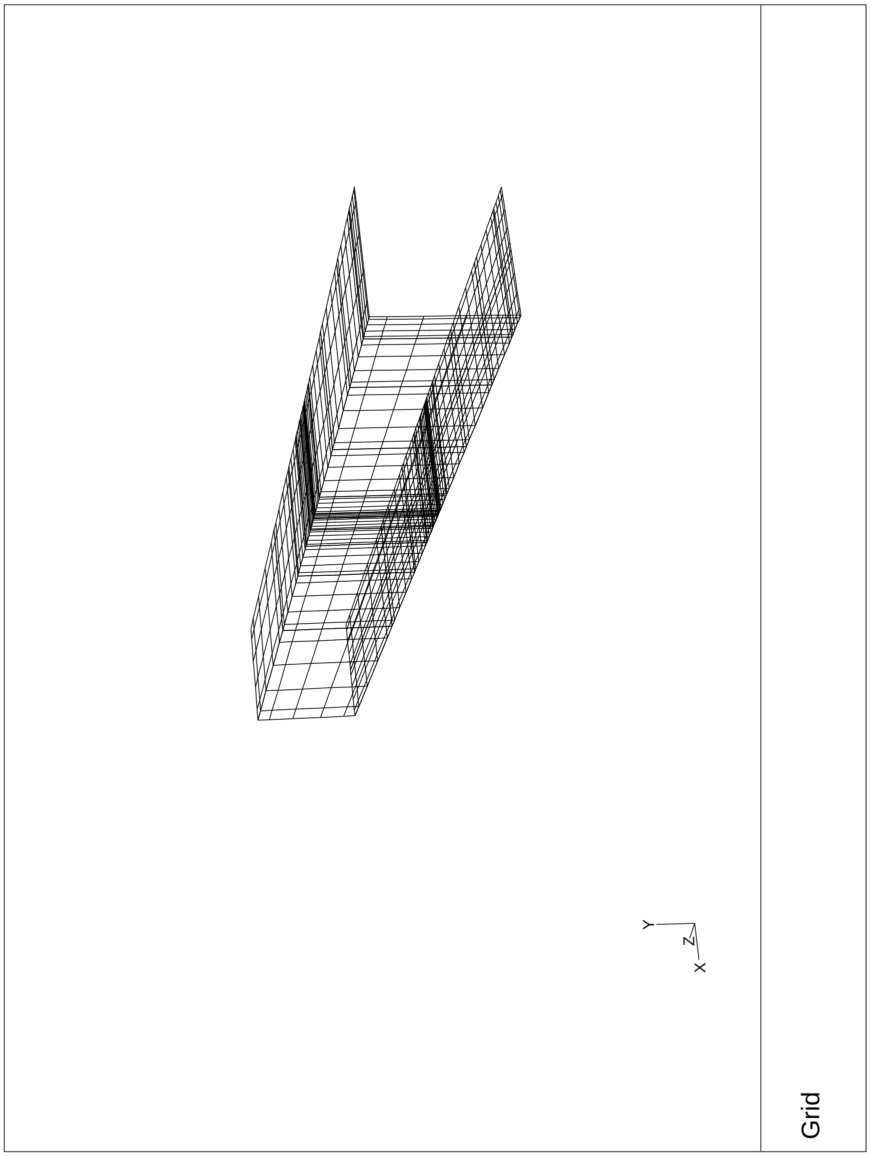


Figure 1 Extrusion of a square die; initial mesh

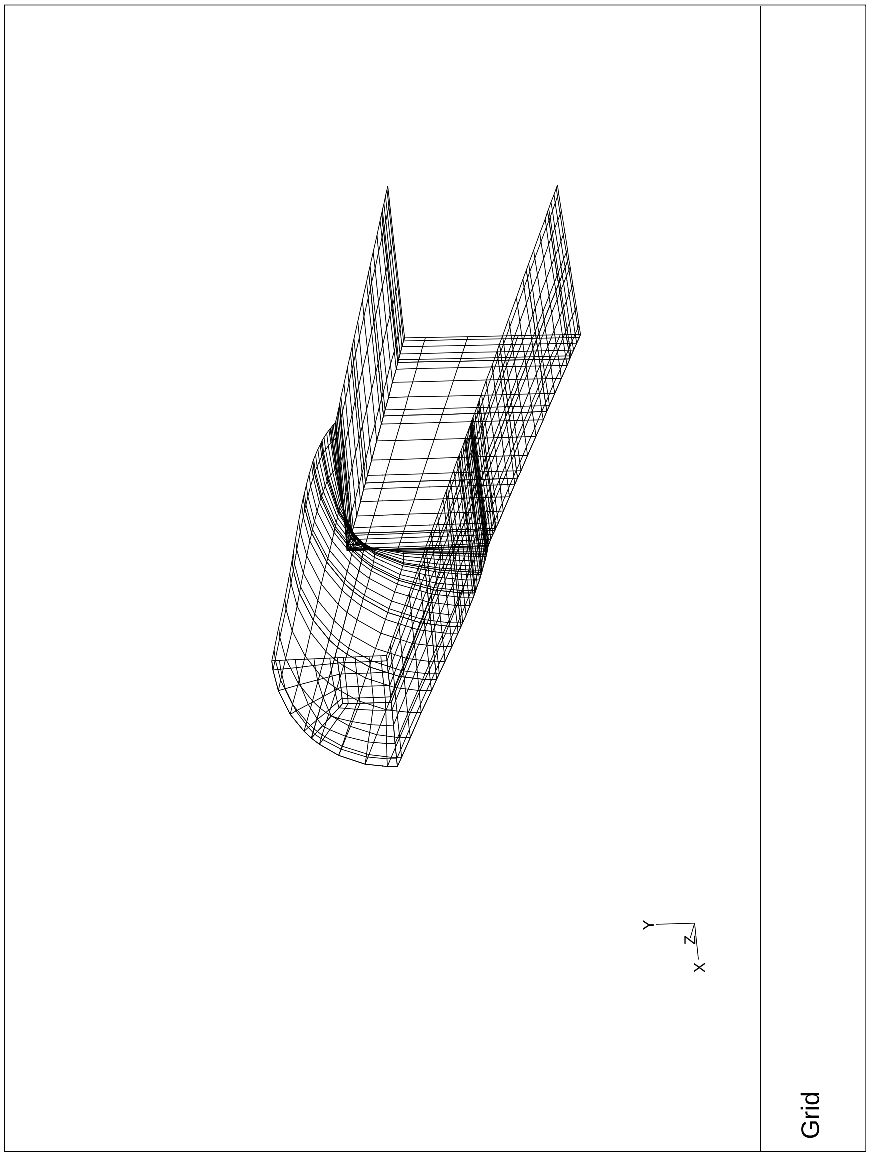


Figure 2 Extrusion of a square die; final mesh; Re=20 and We=2

REFERENCES

[Cas96] Casarin M. (1996) *Schwarz preconditioners for spectral and mortar finite element methods with applications to incompressible fluids.* PhD thesis, New York University.

[CMBBC91] Chenot J., Montmitonnet P., Bern A., and Bertrand-Corsini C. (1991) A method for determining free surfaces in steady state finite element computations. *Computer Methods in Applied Mechanics and Engineering* 92: 245–260.

[DM90] Deville M. and Mund E. (1990) Finite-element preconditioning for pseudospectral solutions of elliptic problems. *SIAM J. Sci. Stat. Comput.* 11(2): 311–342.

[DW90] Dryja M. and Widlund O. (1990) Towards a unified theory of domain decomposition algorithms for elliptic problems. In Chan T., Glowinski R., Periaux J., and Widlund O. (eds) *Proceedings of the Third International Symposium on Domain Decomposition Methods for PDE's.* SIAM.

[EPW92] Ellwood K., Papanastasiou T., and Wilkes J. (1992) Three-dimensional streamlined finite elements: design of extrusion dies. *Int. J. Numer. Meth. in Fluids* 14: 13–24.

[HP91] Ho L. and Patera A. (1991) Variational formulation of three-dimensional viscous free-surface flows: natural imposition of surface tension boundary conditions. *International Journal for Numerical Methods in Fluids* 13: 691–698.

[HR94] Ho L. and Rønquist E. (1994) Spectral element solution of steady incompressible viscous free-surface flows. *Finite Elements in Analysis and Design* 16: 207–227.

[MP89] Maday Y. and Patera A. (1989) Spectral element methods for the Navier-Stokes equations. In Noor A. (ed) *State of the Art Surveys in Computational Mechanics*, pages 71–143. ASME, New York.

[MPR92] Maday Y., Patera A., and Rønquist E. (1992) The $P_N \times P_{N-2}$ method for the approximation of the Stokes problem. Technical Report 92009, Department of Mechanical Engineering, Massachusetts Institute of Technology.

[RE96] Ramanan N. and Engelman M. (1996) An algorithm for simulation of steady free surface flows. *Int. J. Numer. Meth. in Fluids* 22: 103–120.

[Røn96] Rønquist E. (1996) A domain decomposition solver for the steady Navier-Stokes equations. In Ilin A. and Scott L. (eds) *Proceedings of the Third International Conference on Spectral and High Order Methods.* Houston Journal of Mathematics. Conference held in Houston, Texas, 5-9 June 1995.

[SBG96] Smith B. F., Bjørstad P., and Gropp W. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press.

94

Mechanical Criteria for Decomposition into Subdomains

Damien Soulat and Francois Devries

1 Introduction

Domain decomposition methods are widely used in several mechanical applications as, for example, non-linear elasticity or dynamic problems; we propose here, to extend their application to the study of heterogeneous structures. In the next section, we describe a methodology which uses homogenization techniques and subdomain decomposition methods to simulate behavior of composite material (which are strongly heterogeneous structures) and we shall see how it may conduct to good time savings on parallel computers. To this end, we will describe the difficulties encountered, which essentially consist in choosing a decomposition taking into account some mechanical criteria as the presence of heterogeneities. In the subsequent section, starting from a simple example concerning an elastic heterogeneous structure, we shall illustrate the influence of the decomposition, and specially the 'corner's problem' on the efficiency of the Schur complement method.

2 Review of Composite Materials and Homogenization Techniques

Composite materials are becoming more and more important in the construction of high-performances mechanical structures, as for example aerospace applications. Such applications necessarily require a good knowledge of the composite's material properties. The main difficulty in this area stems from the high level of heterogeneity encountered, making any numerical computation prohibitive if not impossible. A way to overcome these difficulties consists in using homogenization technique for periodic structures [Duv76, L8́4].

This technique first consists in considering two scales: the microscopic (connected to the composite basic cell) describes the composite's constituents and the macroscopic relates to the scale of the composite material studied. By a microscopic approach, this technique allows one to compute the equivalent homogeneous behavior for the composite. The computation of all moduli describing this homogeneous behavior

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

is carried out by the solution of problems to be solved at the microscopic level (called cellular problems), whose number and complexity depend on the composite's constituents. The main advantage of this theory is that the computation of the response of a homogeneous structure may be carried out without numerical overcosts. Let us note that this theory has led to the elaboration of many softwares and their efficiencies have been proved many times for several constituent's behaviors.

However the analyze [Dev92] of the CPU times required for this homogenization procedure reveals that 50 percent, is dedicated to the solution of the cellular problems. The first reason to this fact stems from the multiplicity of the cellular problems (for example in linear elasticity 6 cellular problems have to be solved, in order to obtain all moduli of the homogeneous behavior); the second reason consists in the presence of periodicity boundary conditions which affect the bandwidth of the FEM matrix. Thus, when concepters look for the optimal conception of composite materials, because they have to consider a lot of mechanical (moduli of the constituent's behavior)and geometric parameters (volume of inclusions, porosity part,...) to describe the microscopic level for quantifying their effects on the equivalent behavior moduli, it is necessary to multiply by a lot of parameters number the number of the cellular problems to be solved and it is clear that this conception step involves big difficulties when sequential computers are used.

Homogenization Process and Domain Decomposition Method.

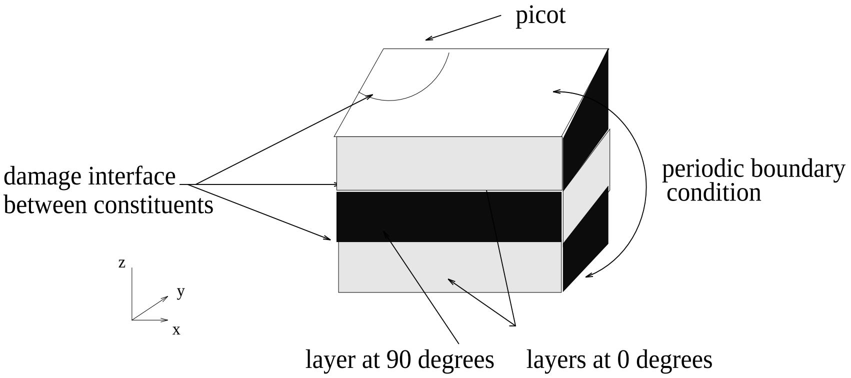
A way to overcome these difficulties, consists in developing [Sou96] a methodology which uses conjointly homogenization techniques and subdomain decomposition methods in order to take advantage in a large way of the parallel computers performance.

To illustrate this methodology, let us consider the case of a thermoelastic composite material, used at industrial level as coating for space engines. It is constituted by unidirectional carbon layers (0/90 degrees) held in contact by a third constituent (carbon) called 'picot'. For this geometry it is shown [L'H96] that the equivalent homogeneous thermoelastic behavior is orthotropic and can be fully computed by the solution of 6 elastic, 3 thermal and 1 coupled cellular problems. Moreover the analyze of stress concentration shows that damages located at the interfaces between constituents must be considered in order to obtain an accurate description of the composite reality. The example studied possess some geometric symmetries in the plane (0,x,y) enabling to carry out the solution of cellular problems only on the quarter of the basic cell (Figure 1). However periodicity boundary conditions between the lower and the upper faces of the quarter of the basic cell, remain. The parallel methodology developed for the computation of the composite damaged equivalent behavior consists in the cellular problems by use solving of a nonoverlapping subdomain decomposition method; we use, here, the Schur complement method [BW86]. To this way, we have to choose a subdomain decomposition for the quarter of the basic cell, accounting the periodicity boundary conditions and the discontinuities arising from damage interfaces.

Decompositions into Subdomains of the Basic Cell

- **Periodic boundary conditions in the decomposition.**

Figure 1 Quarter of the basic cell.


picot
damage interface between constituents
periodic boundary condition
z
y
x
layer at 90 degrees
layers at 0 degrees

If the chosen decomposition is such that faces, where we have to satisfy periodic boundary conditions, belong to two different subdomains, it is then required to prescribe a particular link between these subdomains. Between each point of these faces we have to satisfy the **periodicity of the displacement** and the **antiperiodicity of the stress vector**. These relations being exactly the **same** than these to be satisfied at the interface of the decomposition, the main idea for treating these particular conditions (by a subdomain approach) consists in creating a 'fictive' interface, linking the d.o.f. belonging to faces concerned by periodic boundary conditions, and in adding it to the interface problem which will be classically solved by the conjugate gradient algorithm.

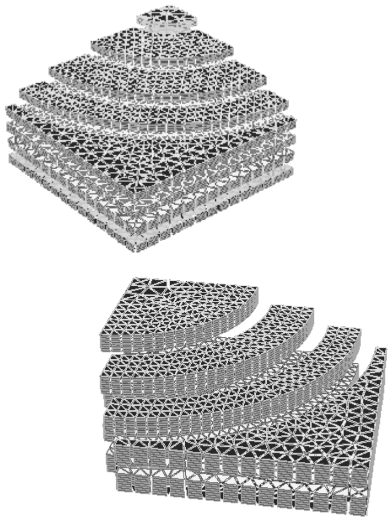
- **Damage interface in the decomposition.**

Two manners to take into account discontinuities in the meshes (modelizing the debonding interfaces) have been considered. They lead to two decompositions in respectively 20 and 8 subdomains (Figure 2). The first one consists in choosing the interfaces of the decomposition as those where damage occurs; when a d.o.f. comes in a damaged zone,(where no relations of continuity have to be satisfied), it is then picked out from the interface problem (decomposition in 20 subdomains). The second manner does not contain this association between damage and decomposition interfaces, and discontinuities arising from damage can be located inside subdomains (decomposition in 8 subdomains).

Results on the KSR Computer

We report (Figure 3) for each decomposition some results obtained during sequential process and parallel process (where the computing tasks related to each subdomain are carried out by each processor assigned). These results concern the solution of one cellular problem by use of the Schur complement method where the two 'Neumann' preconditioners have been considered ('Neumann-Neumann' [LeT94] and 'Neumann-coarse' [Man94]). They have been obtained on the KSR parallel machine of the INRIA institut. We dissociate in these curves the 'local operations' in the subdomains (computations of 'Dirichlet' and 'Neumann' problems) from the 'interface operations'

Figure 2 Decompositions in 20 et 8 subdomains of the quarter of the basic cell



(conjugate gradient operations). These results show that the decomposition in 20 subdomains is not satisfying because during the parallel process, most of time is consumed by the 'interface operations' in opposition to the decomposition in 8 subdomains for which the time saving may be increased by using more than one processor by subdomains. However, because we have chosen a damage located at the interface between the 'picot' (whose radius is small) and the layers, we have associated, for the decomposition in 20 subdomains, only one subdomain for the 'picot'. Thus we have spoiled the load balancing between processors. The decomposition in 8 subdomains does not present this phenomenon since it satisfies some mechanical criteria and a good size equilibrium between subdomains for a parallel application. With this example we raise the problem posed by the influence of the choice of the decomposition on times savings obtained on parallel process. This aspect can be more fully described thanks to the following example which concerns the study of the rate of convergence of domain decomposition methods when applied to heterogeneous structures where 'corners' exist.

3 Corner's Problem for Heterogeneous Structures

The use of the preconditioned conjugate gradient for the solution of the interface problem involves that the rate of convergence (and consequently the times savings) depends on the choice of the decomposition in subdomains ([SV95]), and recent researchs (see [FR94]) try to minimize this influence. To illustrate this aspect, let us consider the problem posed on the multilayered structure and describe in the Figure 4 (where σ is the stress tensor, $\epsilon(u))$ the linearized strain tensor, ν Poisson ratio and E the Young modulus). We study two material configurations: the first, is

Figure 3 Times repartition between operations of the Schur complement method

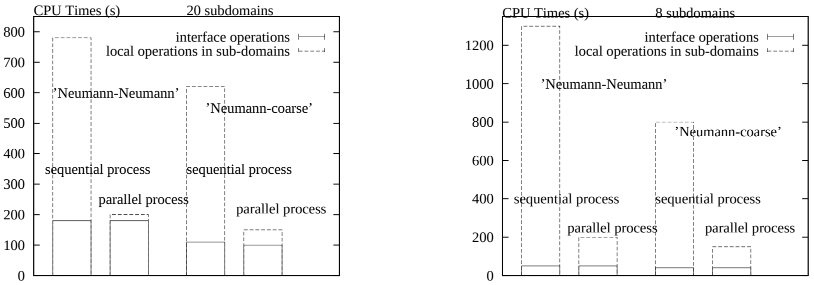
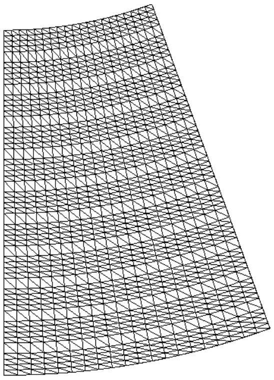


Figure 4 Problem posed on a multilayered structure.



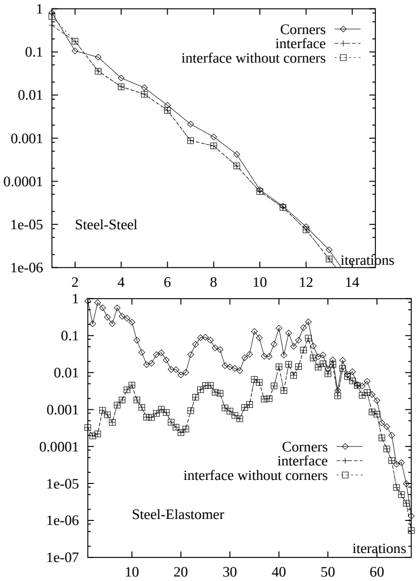
$$div\sigma = 0$$
$$\epsilon_{ij}(u) = \frac{1+\nu}{E}\sigma_{ij} - \frac{\nu}{E}\sigma_{kk}\delta_{ij}$$
$$\epsilon(u) = \frac{1}{2}(\nabla(u) + \nabla^T(u))$$
$u = 0$ on the lower face
$u = U$ on the upper face

homogeneous where all layer are made up of steel (called 'Steel-Steel'); the second is heterogeneous because the layers are alternatively made up of steel and elastomer (called 'Steel-Elastomer'). Four decompositions in 4, 6, 8 and 16 subdomains have been generated. They respect a good load balancing. Moreover some points of subdomain meshes belong to more than 2 subdomains (these points are called 'corners'). For each decomposition, we describe in Table 1.1 the rate of convergence (in iteration's number) of the Schur complement method used with the 'Neumann-coarse' preconditioner. Our purpose here is just to establish that in one case (homogeneous) we obtain a good efficiency of the 'Neumann-coarse' preconditioner as the subdomain number grows, whereas in the other case (heterogeneous) **with the same decomposition** we do not obtained it. To explain this phenomenon, we reporte in figure 1.5 the evolution of the conjugate gradient residual for each point of the interface (for the decomposition in 6 subdomains) and for the two material configurations studied. In these curves,

Table 1 Rate of convergence of the Schur complement method.

number of subdomains	number of corners	iterations with ”Neumann-coarse”
		Steel-Steel — Steel-Elastomer
4	1	12 — 35
6	2	13 — 67
8	3	13 — 84
16	9	16 — 114

Figure 5 Residual of interface points, decomposition in 6 subdomains.



we dissociate in the global interface d.o.f(denoted by ’interface’) the ’corners’ d.o.f (denoted by ’corners’) from the rest of d.o.f. (denoted by ’interface without corners’). It appears that in the heterogeneous case, the residual at the corner points is not decreasing like at the other interface points. We observe the same phenomenon for the other decompositions ([Sou96]). Thus the explanation of the deficiency of this preconditioner stems from the existence of corner points, which require (like in the case of the ’Two-level FETI Method’ ([FM96])) the development of an adequate procedure for their treatments.

4 Conclusion

We have tried to pointed out with the examples studied, the problem encountered by the choice of decomposition in subdomains, when strongly heterogeneous structures are considered. Let us note that heterogeneities could be treated, however, by the coarsening operator [FR95], during the preconditioning step, but the decomposition step remains necessary and requires attention as we saw it with the example of the multilayered structure. However we have shown with the study on composite material that it is possible to use the interface relations to take into account some mechanical criteria as periodicity boundary conditions or the presence of damage interfaces.

REFERENCES

[BW86] Bjorstad P. and Wildlund O. (december 1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SINUM* 23(6): pp. 1097–1120.

[Dev92] Devries F. (July 1992) Modélisation et calcul de structures composites. Habilitation à diriger des recherches, Université Paris 6.

[Duv76] Duvaut G. (September 1976) Analyse fonctionnelle et mécanique des milieux continus. application à l'étude des matériaux composites élastiques à structures périodiques. In KOITER W. (ed) *Theoretical and applied mechanics.* North-Holland.

[FM96] Farhat C. and Mandel J. (1996) The two-level feti method for static and dynamic plate problems - part1. *CMAME* preprint.

[FR94] Farhat C. and Roux F. (Juin 1994) *Implicit parallel processing in structural mechanics*, volume 2 of *Computational Mechanics Advances.* North-Holland.

[FR95] Farhat C. and Rixen D. (July 1995) A new coarsening operator for the optimal preconditioning of the dual and primal domain decomposition methods. In *Cooper Mountain Conference on Multigrid Methods.*

[L84] Léné F. (July 1984) Contribution à l'étude des matériaux composites et de leur endommagement. Doctorat d'état, Université Paris 6.

[LeT94] LeTallec P. (february 1994) *Domain decomposition methods in computational mechanics*, volume 1 of *Computational Mechanics Advances.* North-Holland.

[L'H96] L'Hostis G. (January 1996) *Contribution à l'étude de matériaux composites thermoélastiques.* Phd thesis, Université Paris 6.

[Man94] Mandel J. (1994) Balancing domain decomposition. *ANM* 6: pp. 1313–1319.

[Sou96] Soulat D. (March 1996) *Méthodes de décompositions de domaines et parallélisme en calcul de structures hétérogènes et composites.* Phd thesis, Université Paris 6.

[SV95] Soulat D. and Vidrascu M. (May 1995) Prise en compte de critères mécaniques et géométriques dans le choix de la décomposition en sous-domaines. In Hermes (ed) *Deuxieme Colloque National en Calcul de Structures*, volume 2, pages pp. 711–716.

95

A Quasi-Exact Interface Condition for Implicit Multiblock Euler and Navier-Stokes Calculations

Guy De Spiegeleer and Alain Lerat

1 Introduction

In most domain decomposition methods for Computational Fluid Dynamics, the interface conditions are treated explicitly. When using implicit schemes, this causes stability or efficiency limitations (see [Rai86, Jen94]). For implicit schemes leading to the solution of block-tridiagonal linear systems, one can use the parallel factorisation technique developed in [Wan81] or in [Bru91]. However, this technique needs the sequential computation of a reduced system, which slightly degrades the parallelisation rate of the algorithm.

In this paper, we introduce a completely parallelizable algorithm which yields accurate interface conditions at a quite low cost. It is deduced from the Schwarz alternating method with a fictitious overlapping mesh [Lio88]. We transform this algorithm into another much more efficient one since the latter is equivalent to a one-block overlapping Schwarz algorithm. It allows a strict parallelization because all computations are done independently in each block. Communications are limited to neighbouring blocks. The efficiency of the multiblock implicit solver is assessed by numerical calculations of inviscid and viscous compressible flows around a NACA0012 airfoil.

2 Numerical Scheme

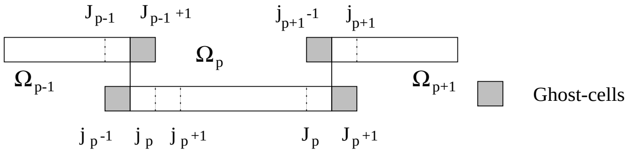
We consider the one-dimensional conservation law system :

$$w_t + f_x = 0, \qquad x \in \Omega \subset \mathbb{R} \tag{2.1}$$

where $w(x,t) \in \mathbb{R}^m$ denotes the vector of conservative variables and f is the flux vector. The flux Jacobian matrix $A = df/dw$ has real eigenvalues and a complete set of eigenvectors.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Figure 1 Multiblock mesh and ghost-cells definition.



Implicit Single-Block Scheme

The numerical solution is denoted by $w_j^n \approx w\left(j\delta x, n\Delta t\right)$, for $j \in \mathcal{J}_0 = \{j_0, \ldots, J_0\}$, Δt and δx are respectively the time-step and the space-increment, $\sigma = \Delta t/\delta x$. System (2.1) is approximated by a three-point implicit scheme which involves two time levels. The explicit stage reads :

$$\Delta w_j^{exp} = -\sigma\left[h\left(w_{j+1}, w_j\right) - h\left(w_j, w_{j-1}\right)\right],\ j \in \mathcal{J}_0$$

where h is the numerical flux associated with f. Note that without significant change, the explicit stage could involve more than three points. The linear implicit stage is given by :

$$n_j^- \Delta w_{j-1} + n_j^0 \Delta w_j + n_j^+ \Delta w_{j+1} = \Delta w_j^{exp},\ j \in \mathcal{J}_0 \tag{2.2}$$

where $\Delta w_j = w_j^{n+1} - w_j^n$. The $m \times m$ matrix blocks n_j^-, n_j^0 and n_j^+ depend on w_{j-1}^n, w_j^n and w_{j+1}^n. The single-block computation of the implicit stage consists in solving the $(J_0 - j_0 + 1) \times (J_0 - j_0 + 1)$ block-tridiagonal matrix $\mathcal{N}_0 = [n_j^-, n_j^0, n_j^+]$, with the Thomas algorithm. Here the vector values Δw_{j_0-1} and Δw_{J_0+1} are assumed to be null for convenience.

Implicit Multiblock Scheme

The computational domain Ω is partitioned into P blocks Ω_p, $p \in \{1, \ldots, P\}$. The block Ω_p is composed of adjacent mesh-cells c_j, $j \in \mathcal{J}_p = \{j_p, \ldots, J_p\}$. A fictitious overlapping is introduced so that the ghost-cells c_{j_p-1} and c_{J_p+1} respectively overlap the interior cells $c_{J_{p-1}}$ and $c_{j_{p+1}}$ (see Fig. 1).

In order to ensure the solution uniqueness, we need interface conditions at the ghost-cells c_{j_p-1} and c_{J_p+1}. The definition of the exact explicit interface condition easily comes from the identification of the multiblock computation using ghost-cells with the equivalent single-block computation. At $j = j_p$ we have for the single-block computation :

$$\Delta w_{j_p}^{exp} = -\sigma\left[h\left(w_{j_p+1}, w_{j_p}\right) - h\left(w_{j_p}, w_{j_p-1}\right)\right]$$

and for the multiblock computation :

$$\Delta w_{j_p}^{exp} = -\sigma\left[h\left(w_{j_p+1}, w_{j_p}\right) - h\left(w_{j_p}, w_{J_{p-1}}\right)\right]$$

Identifying at $j = J_p$, we deduce the exact explicit interface condition :

$$w_{j_p-1}^n = w_{J_{p-1}}^n \text{ and } w_{J_p+1}^n = w_{j_{p+1}}^n, \qquad p \in \{1, \ldots, P\} \tag{2.3}$$

In the same way, we obtain the exact implicit interface condition :

$$\Delta w_{j_p-1} = \Delta w_{J_{p-1}} \text{ and } \Delta w_{J_p+1} = \Delta w_{j_{p+1}}, \qquad p \in \{1, \ldots, P\} \tag{2.4}$$

The implicit interface condition (2.4) couples the implicit stage systems of the P blocks. Consequently, we cannot compute the implicit scheme independently in each block. To uncouple the systems, one solution consists in lagging in time the interface values on the adjacent block so that the interface condition becomes :

$$\left.\begin{array}{rcl} w^n_{j_p-1} = w^{n-1}_{J_{p-1}} & \text{and} & w^n_{J_p+1} = w^{n-1}_{j_{p+1}} \\ \Delta w_{j_p-1} = \Delta w^{n-1}_{J_{p-1}} & \text{and} & \Delta w_{J_p+1} = \Delta w^{n-1}_{j_{p+1}} \end{array}\right. , \qquad p \in \{1, \ldots, P\} \tag{2.5}$$

where $\Delta w^{n-1} = w^n - w^{n-1}$. This time-lagging condition has been proved not to jeopardize the linear stability of the interface problem [LW96]. However, the interface condition (2.5) causes a partition dependency which can degrade the solver efficiency for a large number of blocks.

3 New Implicit Interface Condition

The reduced system

The construction of the implicit interface condition proposed in this paper begins like a classical parallelizable tridiagonal-linear system solver. We compute the influence of the ghost-cells on the interior-cell time-increments. The implicit multiblock scheme using ghost-cells is written in each block $p \in \{1, \ldots, P\}$ as :

$$\underbrace{\begin{bmatrix} n^0_{j_p} & n^+_{j_p} & & \\ n^-_{j_p+1} & \ddots & \ddots & \\ & \ddots & \ddots & n^+_{J_p-1} \\ & & n^-_{J_p} & n^0_{J_p} \end{bmatrix}}_{\mathcal{N}_p} \begin{bmatrix} \Delta w_{j_p} \\ \Delta w_{j_p+1} \\ \vdots \\ \Delta w_{J_p-1} \\ \Delta w_{J_p} \end{bmatrix} = \begin{bmatrix} \Delta w^{exp}_{j_p} \\ \Delta w^{exp}_{j_p+1} \\ \vdots \\ \Delta w^{exp}_{J_p-1} \\ \Delta w^{exp}_{J_p} \end{bmatrix} - \begin{bmatrix} n^-_{j_p} \Delta w_{j_p-1} \\ 0 \\ \vdots \\ 0 \\ n^+_{J_p} \Delta w_{J_p+1} \end{bmatrix} \tag{3.6}$$

The vector values Δw_{j_1-1} and Δw_{J_P+1} are assumed to be null. The inversion of the matrix $\mathcal{N}_p$ shows that the linear system (3.6) is a linear form which links the ghost-cell values Δw_{j_p-1} and Δw_{J_p+1} to the time-increments Δw_j, $j \in \{j_p, \ldots, J_p\}$:

$$\Delta w_j = \Delta \overline{w}_j + B^-_j \Delta w_{j_p-1} + B^+_j \Delta w_{J_p+1}, \qquad j \in \mathcal{J}_p \tag{3.7}$$

where $\Delta \overline{w}_j$ is the numerical value of the time-increment in Ω_p assuming a zero value at each interface, and the matrices B^-_j and B^+_j are the *influence matrices.* The computation of $\Delta \overline{w}_j$, B^-_j and B^+_j only depends on the interior data in Ω_p. They are computed independently in each block using the following algorithm adapted from the Thomas algorithm.

```
$U_{j_p-1} = 0$
$C_{j_p-1} = I$
$\overline{y}_{j_p-1} = 0$
DO $j = j_p, J_p$
  $D_j = -n_j^- U_{j-1} + n_j^0$
  $U_j = (D_j)^{-1} n_j^+$
  $C_j = -(D_j)^{-1} n_j^- C_{j-1}$
  $\overline{y}_j = (D_j)^{-1} \left(\Delta w_j^{exp} - n_j^- \overline{y}_{j-1}\right)$
ENDDO
$\Delta\overline{w}_{J_p} = \overline{y}_{J_p}$
$B_{J_p}^- = C_{J_p}$
$B_{J_p}^+ = -U_{J_p}$
DO $j = J_p - 1, j_p$
  $\Delta\overline{w}_j = \overline{y}_j - U_j \Delta\overline{w}_{j+1}$
  $B_j^- = C_j - U_j B_{j+1}^-$
  $B_j^+ = -U_j B_{j+1}^+$
ENDDO
```

Now we assemble the interface problem (or *reduced system* or *global Schur complement operator*) whose unknowns are the ghost-cell values. It corresponds to the linear form (3.7) for $j = J_1, j_2, \ldots, j_p, J_p, \ldots, J_{P-1}, j_P$, where the exact implicit interface condition (2.4) has been applied :

$$\left\{\begin{array}{llll}
\Delta w_{J_1} & = \Delta\overline{w}_{J_1} & & +B_{J_1}^+ \Delta w_{j_2} \\
\Delta w_{j_2} & = \Delta\overline{w}_{j_2} & +B_{j_2}^- \Delta w_{J_1} & +B_{j_2}^+ \Delta w_{j_3} \\
 & \ldots & & \\
\Delta w_{j_p} & = \Delta\overline{w}_{j_p} & +B_{j_p}^- \Delta w_{J_{p-1}} & +B_{j_p}^+ \Delta w_{j_{p+1}} \\
\Delta w_{J_p} & = \Delta\overline{w}_{J_p} & +B_{J_p}^- \Delta w_{J_{p-1}} & +B_{J_p}^+ \Delta w_{j_{p+1}} \\
 & \ldots & & \\
\Delta w_{J_{P-1}} & = \Delta\overline{w}_{J_{P-1}} & +B_{J_{P-1}}^- \Delta w_{J_{P-2}} & +B_{J_{P-1}}^+ \Delta w_{j_P} \\
\Delta w_{j_P} & = \Delta\overline{w}_{j_P} & +B_{j_P}^- \Delta w_{J_{p-1}} &
\end{array}\right. \tag{3.8}$$

This reduced system has $2(P-1)$ equations and $2(P-1)$ unknowns. It can be solved with the Wang's direct algorithm [Wan81]. But this algorithm is only efficient when the cell number in each block is much larger than the block number.

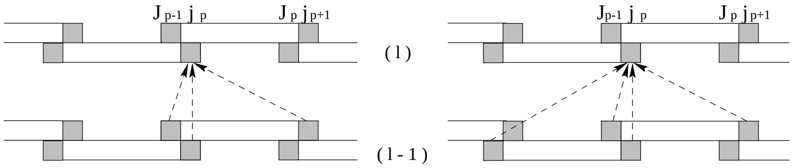
The iterative process

Taking advantage of the particular structure of our problem, we have easily formed the reduced system. Thus we can completely relax system (3.8) by introducing the following iterative algorithm :

```
Initialization of $\Delta w_{j_p}^{(0)} = 0$ and $\Delta w_{J_p}^{(0)} = 0$
DO $l = 1, L$
    $\Delta w_{j_p}^{(l)} = \Delta \overline{w}_{j_p} + B_{j_p}^{-} \Delta w_{J_{p-1}}^{(l-1)} + B_{j_p}^{+} \Delta w_{j_{p+1}}^{(l-1)}$
    $\Delta w_{J_p}^{(l)} = \Delta \overline{w}_{J_p} + B_{J_p}^{-} \Delta w_{J_{p-1}}^{(l-1)} + B_{J_p}^{+} \Delta w_{j_{p+1}}^{(l-1)}$
ENDDO
$\Delta w_{j_p - 1} = \Delta w_{J_{p-1}}^{(L)}, \ \Delta w_{J_p + 1} = \Delta w_{j_{p+1}}^{(L)}$
```

This algorithm consumes very little CPU time because an iteration represents only four matrix-vector multiplications independently computed in each block. Nevertheless, its convergence rate is proportional to the overlapping size. On Fig. 2 -left, we represent by an arrow the geometrical dependency of the ghost-cells from iteration $(l-1)$ to iteration (l). Since it is equivalent to a one-cell overlapping Schwarz algorithm, this algorithm requires many iterations to reach a perfect accuracy.

Figure 2 Left : Slow iterative algorithm. Right : Fast iterative algorithm.



In order to dramatically increase this low efficiency, we suggest a second algorithm. We now relax system (3.8) according to the following relation :

$$\begin{array}{llll} \Delta w_{j_p}^{(l)} & = \Delta \overline{w}_{j_p} & + B_{j_p}^{-} \Delta w_{J_{p-1}}^{(l)} & + B_{j_p}^{+} \Delta w_{j_{p+1}}^{(l-1)} \\ \Delta w_{J_p}^{(l)} & = \Delta \overline{w}_{J_p} & + B_{J_p}^{-} \Delta w_{J_{p-1}}^{(l-1)} & + B_{J_p}^{+} \Delta w_{j_{P+1}}^{(l)} \end{array} , \qquad p \in \{1, \ldots, P\}$$

We find our new implicit interface algorithm :

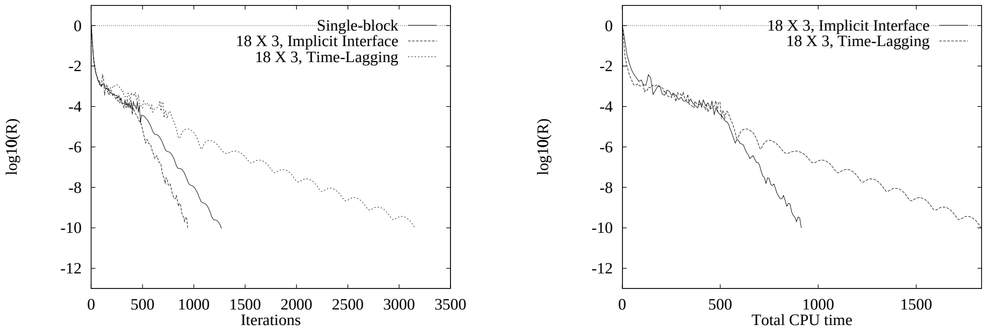
```
Initialization of $\Delta w_{j_p}^{(0)} = 0$ and $\Delta w_{J_p}^{(0)} = 0$
DO $l = 1, L$
    $\Delta w_{j_p}^{(l)} = \Delta \overline{\overline{w}}_{j_p} + \overline{B}_{j_p}^{-} \Delta w_{J_{p-2}}^{(l-1)} + \overline{B}_{j_p}^{+} \Delta w_{j_{p+1}}^{(l-1)}$
    $\Delta w_{J_p}^{(l)} = \Delta \overline{\overline{w}}_{J_p} + \overline{B}_{J_p}^{-} \Delta w_{J_{p-1}}^{(l-1)} + \overline{B}_{J_p}^{+} \Delta w_{j_{p+2}}^{(l-1)}$
ENDDO
$\Delta w_{j_p - 1} = \Delta w_{J_{p-1}}^{(L)}, \ \Delta w_{J_p + 1} = \Delta w_{j_{p+1}}^{(L)}$
```

where the new influence matrices are computed in parallel before the first iteration as :

$$\left| \begin{array}{ll} \Delta \overline{\overline{w}}_{j_p} & = M_{j_p} \left(\Delta \overline{w}_{j_p} + B_{j_p}^{-} \Delta \overline{w}_{J_{p-1}} \right) \\ \overline{B}_{j_p}^{-} & = M_{j_p} B_{j_p}^{-} B_{J_{p-1}}^{-} \\ \overline{B}_{j_p}^{+} & = M_{j_p} B_{j_p}^{+} \end{array} \right. \qquad \left| \begin{array}{ll} \Delta \overline{\overline{w}}_{J_p} & = M_{J_p} \left(\Delta \overline{w}_{J_p} + B_{J_p}^{+} \Delta \overline{w}_{j_{p+1}} \right) \\ \overline{B}_{J_p}^{-} & = M_{J_p} B_{J_p}^{-} \\ \overline{B}_{J_p}^{+} & = M_{J_p} B_{J_p}^{+} B_{j_{p+1}}^{+} \end{array} \right.$$

$$\text{with } M_{j_p} = \left(I - B_{j_p}^{-} B_{J_{p-1}}^{+} \right)^{-1} \qquad \text{with } M_{J_p} = \left(I - B_{J_p}^{+} B_{j_{p+1}}^{-} \right)^{-1}$$

Figure 3 Inviscid transonic flow ($M_\infty = 0.85$, $\alpha = 1^o$). Convergence history in terms of iterations and CPU time.



This iterative process is now equivalent to a one-block overlapping Schwarz algorithm (see Fig. 2 -right). The convergence rate of the latter algorithm (called the fast algorithm) is much greater than the convergence rate of the previous one because the overlapping distance has been increased from one cell to one block. It offsets the increasing cost due to the computation of the new influence matrices.

4 Numerical Applications

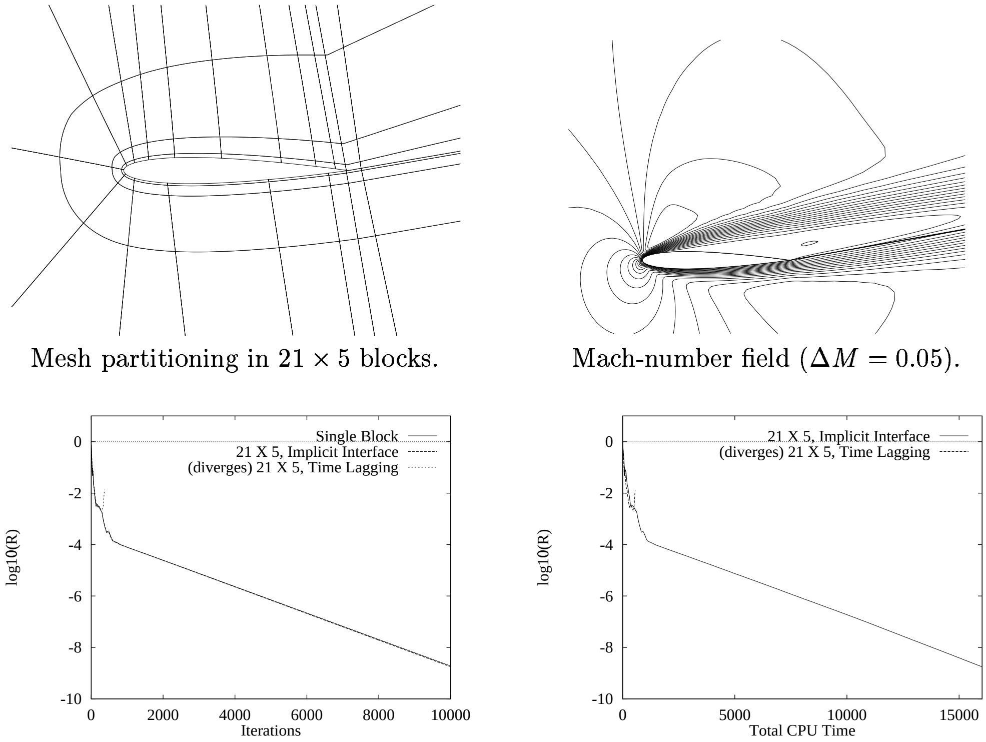
For the numerical applications, we use the centered implicit dissipative Lerat scheme [Ler83]. This inviscid method is second-order accurate, unconditionally stable and dissipative in the sense of Kreiss. For viscous flow computations, we use an extension due to Hollanders *et al.* [HLP83]. The implicit system is solved using a line Jacobi relaxation procedure so that the computation of the two-dimensional scheme is equivalent to two computations of the one-dimensional scheme (2.2). A local time step is used so that the CFL number is uniform all over the mesh. The computations start from an uniform flow. All computations have been made without artificial viscosity.

The present multiblock implicit solver has been implemented on a Cray J916. Its efficiency is assessed by computing two external steady flows around a NACA 0012 airfoil and comparing the convergence histories of the single and multiblock methods using either the time-lagging interface condition or the new implicit interface condition with only two iterations of the fast iterative algorithm. The comparison is made in terms of iterations and also of CPU times for the multiblock solvers.

Inviscid transonic flow

The first application is an inviscid transonic flow at freestream Mach number $M_\infty = 0.85$ with a 1^o angle of attack. The calculation is run on a 188×24 C-mesh which is partitioned into $18 \times 3 = 54$ blocks of approximately 10×8 cells each, the CFL number is equal to 20.

Figure 4 Viscous subsonic flow ($M_\infty = 0.8$, $\alpha = 10^o$). Convergence history in terms of iterations and CPU time.



Mesh partitioning in 21 × 5 blocks.

Mach-number field ($\Delta M = 0.05$).

The convergence histories are shown on Fig. 1.3. In terms of time iterations, all the methods behave similarly at the beginning of the transient phase, but after 500 iterations the implicit interface condition is much more efficient than the time lagging condition. The number of iterations to reach a L_2 - residual of 10^{-10} is reduced by a factor 3.3. The CPU-time per iteration depends on the coding and the load balancing between the explicit and implicit stage of the scheme. The extra-cost per iteration (over time-lagging) decreases when the flow modelization is improved and varies from 65 % for the present Euler computation to 5% for a turbulent Navier-Stokes computation. Therefore, in the present Euler test-case, the total CPU-time to reach the 10^{-10} residual is reduced by a factor 2 (see Fig. 1.3).

Furthermore, Fig. 1.3-left shows that the implicit multiblock solver converges faster than the single block solver. This surprising result is due to a better treatment of the cut line issuing from the trailing edge. Actually the matching is time lagged in the single block computation while it is accurately implicited in the proposed multiblock treatment.

Viscous subsonic flow

The second application is a viscous subsonic laminar flow ($Re = 500$) at free stream Mach number $M_\infty = 0.8$ with a 10^o angle of attack. The C-mesh is composed of 254×58 cells. It is partitioned into $21 \times 5 = 105$ blocks of roughly 12×12 cells (see Fig. 4). The CFL number is equal to 35. On the Mach number field, we observe a large recirculation region.

The multiblock solver using the time-lagging interface condition diverges while the use of the new implicit interface condition yields the single-block solver efficiency. Note that even if there are numerous blocks into the boundary layer, no numerical problem has been found with for the multiblock calculation.

5 Conclusion

A new implicit interface condition has been developed for the solution of the Euler and the Navier-Stokes equations on multiblock structured meshes. This method is perfectly parallelizable and as accurate as an exact implicit interface condition.

The present method reduces the CPU cost of a steady flow computation with respect to a block-Jacobi solver or simply allows the convergence when the latter is not able to reach the steady-state.

REFERENCES

[Bru91] Brugnano L. (1991) A parallel solver for tridiagonal linear systems for distributed memory parallel computers. *Par. Comput.* 5: 1017–1023.

[HLP83] Hollanders H., Lerat A., and Peyret R. (1983) Three-dimensional calculation of transonic viscous flows by an implicit method. *AIAA P.* 83-1953. (1985) *AIAA J.* 23 : 1670–1678.

[Jen94] Jenssen C. B. (1994) Implicit multiblock Euler and Navier-Stokes calculations. *AIAA J.* 32(9): 1808–1814.

[Ler83] Lerat A. (1983) Implicit methods of second order accuracy for the Euler equations. *AIAA P.* 83-1925. (1985) *AIAA J.* 23: 33–40.

[Lio88] Lions P. L. (1988) On the Schwarz alternating method I. In Glowinski R., Golub G. H., Meurant G. A., and Périaux J. (eds) *Proc. First Int. Conf. on Domain Decomposition Meths.*, pages 1–42. SIAM, Philadelphia.

[LW96] Lerat A. and Wu Z. N. (1996) Stable conservative multidomain treatments for implicit Euler solvers. *J. Comp. Phys.* 123: 45–64.

[Rai86] Rai M. M. (1986) An implicit, conservative, zonal-boundary scheme for Euler equation calculations. *Comp. Fluids* 14(3): 295–319.

[Wan81] Wang H. (1981) A parallel method for tridiagonal equations. *ACM Trans. Math. Software* 7: 170–183.

96

A Domain Decomposition Strategy for Simulation of Industrial Fluid Flows

Rune Teigland

1 Introduction

Multiblock methods are often employed to compute flows in complex geometries. This paper describes a robust and efficient multiblock solution procedure for general three dimensional flow situations involving both single- and multi-phase flows. The multiblock approach is implemented within the framework of the well known SIMPLE solution strategy. The multiblock linear solvers are based upon acceleration of the basic additive Schwarz method using Krylov subspace methods in the 'outer iterations'. The effect of incorporating the multiblock linear solver within the SIMPLE solution procedure is being discussed in some detail. Test results of the numerical solution and convergence behavior on several problems involving incompressible single- and multiphase flows are presented. The problems chosen involve fearly simple geometries, yet they illustrate the effect of using the multiblock procedure in a general purpose code.

The multiblock approach (in 3D) is to segment the physical region into contiguous subregions, each bounded by six curved sides and each of which transforms to a cubic block in the computational region. Each grid block is assumed to be topologically cubic (in 3D) and has its own curvilinear coordinate system. Furthermore, the use of multiblock grids is advantageous in terms of economic use of memory and the possibility to use different flow equations in different blocks. One should also notice the potential for significant speedup on parallel machines in using the multiblock approach as compared to using a single block approach. Since multiblock grids are unstructured on the block level, information is needed on the connectivity of neighbouring blocks along with their orientation. Each block has its own local coordinate system, needed to provide geometrical flexibility. The multiblock approach can also be used to handle situations involving cyclic boundary conditions, here a block can 'wrap around' the same edge.

The Krylov methods used here are the standard conjugate gradient method (used for solving the pressure correction equation in situations involving incompressible flows)

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

and the BiCGSTAB method proposed in [VdV92]. One should notice that for general compressible flows the arising pressure correction equation is also nonsymmetric. In the case of incompressible flow the pressure correction equation can be solved using the Conjugate Gradient method, an option given in our code. Multiblock approaches to solving viscous fluid flow problems can be found in e.g. [KG93, TF92].

2 Governing Equations and Discretization

The multiblock linear solver presented here has been implemented into a computer program [GST93] for simulation of fluid flows in complex three dimensional geometries. The code is based on the use of general curvilinear coordinates and is applicable to both laminar and turbulent flows as well as multi-phase flows using a model for dispersed flow. It is assumed that the fluid is a Newtonian fluid such that the flow is governed by the Navier-Stokes equations which express conservation of mass and momentum. In the case of turbulent reactive flows the equations are augmented by equations for the conservation of enthalpy, a standard turbulence model as well as equations describing the conservation of mass fraction of a chemical specie. The conservation equations may be written in general form as follows:

$$\frac{\partial(\rho\phi)}{\partial t}+\frac{\partial(\rho u_j\phi)}{\partial x_j}=\frac{\partial}{\partial x_j}(\Gamma_\phi\frac{\partial\phi}{\partial x_j})+S_\phi \tag{2.1}$$

where ϕ represents different conserved quantities such as momentum, continuity, enthalpy, turbulent kinetic energy etc. In the steady state case this is a prototype of a scalar advection-diffusion equation. These equations are discretised on a non-orthogonal co-located grid arrangement. If a flux vector $\boldsymbol{J}_j$ containing convection and diffusion is defined as

$$\boldsymbol{J}_j=\rho u_j\phi-\Gamma_\phi\frac{\partial\phi}{\partial x_j} \tag{2.2}$$

equation (2.1) can be written as

$$\frac{\partial(\rho\phi)}{\partial t}+\frac{\partial\boldsymbol{J}_j}{\partial x_j}=S_\phi \tag{2.3}$$

We use a finite volume discretization scheme where the equations for the conserved quantities are integrated over a general non-orthogonal control volume. Multiblock grids are handled using a local coordinate system. These local coordinate systems may be nontrivially related as block boundaries are crossed, (e.g. orientation change as well as discontinuous grid line slopes as in the example shown in Figure 1). Thus, the discretization of the equations proceeds block by block, exactly as in the single block case. Implicit time discretization (backward Euler) for the transient term yields

$$\int_{\delta V}\frac{\partial(\rho\phi)}{\partial t}=\frac{(\rho_P\phi_P-\rho_P^0\phi_P^0)}{\triangle t}\delta V_P \tag{2.4}$$

where superscript 0 denotes values from the previous time step. The general conservation equation for ϕ, equation (2.1), is integrated over a three-dimensional control volume δV_p in physical space such that after employing Gauss divergence theorem one gets

$$\frac{(\rho_p \phi_p - \rho_p^0 \phi_p^0)}{\triangle t} \delta V_p + \sum_{nn} \boldsymbol{J} \cdot \boldsymbol{A} \mid_{nn} = S_p \tag{2.5}$$

where nn denotes the cell face index, p is the cell-center index, $\boldsymbol{A}$ area vector and S_p is the total source in the control volume. Because the grid is non-orthogonal the derivatives that occur in the viscous and pressure terms must be evaluated in the transformed curvilinear coordinate system (ξ_1, ξ_2, ξ_3). If we use the chain rule for differentiation, we get

$$\frac{\partial \phi}{\partial x_j} = \sum_l \frac{\partial \xi_l}{\partial x_j} \frac{\partial \phi}{\partial \xi_l} = \sum_l e^l \frac{\partial \phi}{\partial \xi_l} \tag{2.6}$$

where e^l are the contravariant basis vectors of the curvilinear coordinates. Details of the derivation can be found in [Mel90, BW87]. In order to alleviate checkerboard oscillations in pressure (due to the use of a co-located grid arrangement) we use a pressure weighted interpolation of the cell-face velocities in the discretised continuity equation. The idea goes back to Rhie and Chow [RC83].

The coupled momentum and continuity equations are solved in a sequential manner using the SIMPLE method of Patankar and Spalding [PS72]. Notice that the arising linear system for each component of the velocity is nonsymmetric. The SIMPLE solution method belongs to the well-known class of pressure correction methods popular in the primitive variable computation of complex incompressible flows [Dec92]. The methods use a predictor-corrector approach to the solution of the Navier-Stokes equations. The SIMPLE method is widely used in general purpose CFD codes. It is a robust methodology applicable for complex, turbulent recirculating flows and it is potentially applicable to all regimes from incompressible laminar flows to supersonic flows. The basic SIMPLE solution strategy consists of iteratively solving the momentum and the pressure correction equations. The solution of scalar equations such as equations for the turbulence production rate and turbulence dissipation rate are then solved using essentially the same basic form of the equations (2.1). The complete main loop in each time step is given by

Algorithm 1 (Main loop)

```
Update primary and derived fields, time,
boundary conditions etc.
Iteration Loop:
   Solve u-velocities for each phase
   Solve v-velocities for each phase
   Solve w-velocities for each phase
   Calculate advective fluxes by Rhie and Chow interpolation
   Solve pressure correction
      Update pressure, velocity, advective fluxes and density
   Solve volume fractions (if number of phases > 1)
repeat (if necessary)
Solve scalar fields
```

When only the steady-state solution is of interest, the time step, $\triangle t$, is used as a parameter through which the convergence rate may be optimized. The viscous flux (see (2.5)) is split in a primary flux that contains the orthogonal terms (relating to grid) which are treated implicitly, while the other terms are treated explicitly (i.e. lumped into the source terms). Thus the arising system of linear equations takes the form

$$a^p \phi^p = \sum_{nb} a^{nb} \phi^{nb} + S_p \tag{2.7}$$

where each point p is coupled to its six neighbours (nb) and S_p denotes the discretised source term. The source term is split such that the resulting matrix associated with the linear system (2.7) is an M-matrix with constant bandwidth. In the approximate inversion of the subdomain matrices on each block an Incomplete Factorization method is being used here. The M-matrix property is important for the existence of the Incomplete Factorization method used as preconditioner in connection with the Krylov subspace methods [Hac94].

3 Multiblock Linear Solver

In this section we present the multiblock linear solvers based upon acceleration of the basic Schwarz alternating method. A thorough discussion and derivation of various domain decomposition algorithms can be found in [SBG96].

The alternating Schwarz method consists of dividing the computational domain into overlapping domains and using efficient solvers on the subdomains. It must be emphasized that the solution of the subproblems can never solve the complete problem on the composite domain, but only represent a partial step of the algorithm. The presentation and derivation of the additive Schwarz method can be cast using the linear algebra framework, see e.g. [SBG96] or [Hac94].

Let the corresponding matrices for the local subproblems be denoted by A_i. We define a (rectangular) restriction matrix R_i such that when applied to a vector x, it gives back the vector of smaller dimension x_i having components that refers to grid points in subdomain Ω_i. Similarly the prolongation operator is defined as the adjoint of the restriction operator (not necessary in general but used here in the derivation). The submatrices A_i corresponding to the local subproblems can either be formed using the same discretization procedure as for the composite problem (this is the route we have chosen) or they can be computed automatically via a so-called Galerkin formulation, see e.g. [Wes92]. The Galerkin (or variational) method of constructing these submatrices is

$$A_i = R_i A R_i^T \tag{3.8}$$

If we define the operator

$$B_i = R_i^T \left(R_i A R_i^T \right)^{-1} R_i \tag{3.9}$$

the Additive Schwarz method can be written as

$$x^{n+1} = x^n + B(b - Ax^n) \tag{3.10}$$

where b denotes the right hand side and A and x are composite grid matrix and solution vector respectively, and n is the iteration number. The preconditioner B is given by

$$B = \sum_i B_i \tag{3.11}$$

The iteration procedure can now be accelerated by using some appropriate Krylov subspace method. The main goal is the solution of the composite grid problem and the coupling between grid blocks has to be accounted for. The coupling is via the preconditioner and the overlap between grid blocks. In the case that meshing in the subregions is identical in the overlapping region the solution procedure simplifies. Interpolation of the variables is then avoided since the overlapping control volumes are made identical to the corresponding internal control volumes. Where the grid blocks are not connected, the extra control volumes are collapsed. The variables will then be located on the sides of the control volumes as before and will define boundary values. In the algorithm below $(\cdot, \cdot)$ denotes the inner product between two vectors.

Algorithm 2 (Multiblock Bi-CGSTAB linear solver)

```
$x_0$ initial guess, $r_0 = B(b - Ax_0)$;
($\bar{r_0}$ arbitrary such that $(\bar{r_0}, r_0) \neq 0$)
$\omega_0 = \rho_0 = \alpha_0 = 1$ and $\nu_0 = p_0 = 0$;
for $i = 1, 2, 3, \ldots$
    $\rho_i = (\bar{r_0}, r_{i-1})$;
    $\beta_{i-1} = (\rho_i / \rho_{i-1})(\alpha_{i-1} / \omega_{i-1})$;
    $p_i = r_i + \beta_{i-1}(p_{i-1} - \omega_{i-1}\nu_{i-1})$;
    $\hat{p} = Bp_i$ ; $\nu_i = A\hat{p}$ ;
(add contributions + overlap, i.e. after $\nu_i = A\hat{p}$ for $\nu_i$ variable).
    $\alpha_i = \rho_i / (\bar{r_0}, \nu_i)$; $s = r_i - \alpha_i \nu_i$;
    if $\|s\|$ small? then $x_{i+1} = x_i + \alpha_i \hat{p}$; exit loop;
    $z = Bs$; $t = Az$;
(add contributions + overlap, i.e. after $t = Az$ for $t$ variable).
    $\omega_i = (t, s)/(t, t)$;
    $x_{i+1} = x_i + \alpha_i \hat{p} + \omega_i z$;
    if $x_i$ accurate enough? then exit loop;
    $r_{i+1} = s - \omega_i t$;
end for
```

4 Multiblock Strategies Combined into the SIMPLE Solution Method

The flow field in all grid blocks is solved using the multiblock linear solvers described in the previous section. Some modifications to the SIMPLE algorithm have to be done.

The following algorithm describes what we call the SIMPLE-Schwarz algorithm, i.e. it uses the multiblock linear solvers described above.

Algorithm 3 (SIMPLE-Schwarz method)

```
Time step loop:
    Update primary and derived fields, time, boundary conditions etc.
    Iteration Loop:
        Solve u-velocities (for all blocks (algorithm 2)) for each phase
        Solve v-velocities (for all blocks (algorithm 2)) for each phase
        Solve w-velocities (for all blocks (algorithm 2)) for each phase
        Calculate advective fluxes by Rhie and Chow interpolation
        Solve pressure correction (for all blocks (algorithm 2))
        Update pressure, velocity, advective fluxes and density
        Solve volume fractions (for all blocks (algorithm 2))
    repeat (if necessary)
    Solve scalar fields (for all blocks (algorithm 2))
Repeat
```

In the steps above all equations are solved in turn over the entire composite region. The updating of variables as well as the calculation of advective fluxes is done for all blocks. An alternative would be to perform the interblock coupling once every time step, i.e. as in the algorithm below

Algorithm 4 (Alternative time stepping loop)

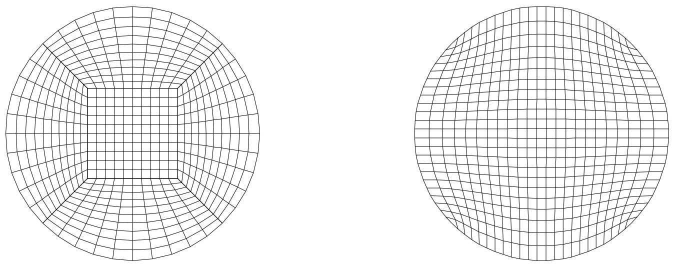
```
Time step loop:
do $i = 1, \dots,$ max. number of blocks
    SIMPLE loop given by algorithm 1
    Update boundary conditions (inner and outer)
end do
```

Most of the routines used need no modification to handle the multiblock situation. Pointers to the different block variables being accessed have to be set up properly. A couple of situations warrant extra precautions. The Rhie and Chow algorithm involves addressing pressure variables at two neighboring points in each direction, and the other situation is when extrapolating variables to boundaries. In situations involving transient incompressible flows or flows involving elliptic regions algorithm 3 is more suitable than algorithm 4 since in those situations the coupling is stronger across block boundaries. Furthermore, in incompressible flows pressure disturbances are felt instantly over the entire region and thus using algorithm 3 seems more appropriate in that case.

5 Numerical Experiments

In this section we present results of a few numerical experiments involving incompressible turbulent flow in a straight pipe section as well as an example involving gravitational sedimentation. Simulation of turbulent flow in pipes is an important field of research, e.g. in the development of multiphase flow metering devices. Preliminary

Figure 1 Cross section of grid, left: 5 block configuration, right: H-type grid.



results with our multiblock approach on some simple laminar flows has been presented previously, see [Tei95].

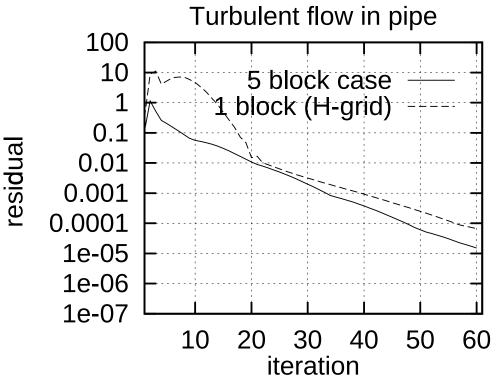
The first set of problems considered here involve 3D turbulent flow in a cylindrical pipe using a five-block configuration and a one block configuration using an H-type grid. In both cases a uniform flow is given at the inlet, and at the outlet variables are extrapolated. The Reynolds number is 32300 and the total number of cells is 10000. The length of the pipe is $20m$ and the radius of the pipe is $1.4m$. The prescribed inlet velocity was $9.9m/s$. This problem was run for a one block H-type grid and a five block grid, see Figure 1.

We notice that in the H-type grid there are 4 corner cells degenerating into triangles. In the figure below we show results of the convergence history of mass residuals versus time iteration with the 5-block and 1-block configuration (H-type grid). We notice that the 5-block configuration is more efficient. H-type grids are widely used in simulation of flow in pipes using a single block approach. We remark that the multiblock approach (5-block case) can easily be parallelized and therefore a significant speedup can potentially be achieved compared to the use of a single block grid (H-type grid).

The last example consists of gravitational sedimentation of an initially homogeneous mixture of a gas and a liquid phase. The densities of the gas and liquid phases are $\rho_{\text{gas}} = 1.2kg/m^3$ and $\rho_{\text{liquid}} = 980kg/m^3$, respectively. The viscosity of the phases are chosen to be in the range of air and water respectively. The computational domain is a channel of dimensions $1m \times 10m$, with free slip boundaries.

The results from computations with the multiblock version of our code is given in the figure below. The domain was divided into a 2×2 grid block system, each block of size $0.5m \times 5m$ each. In the figure below a vertical cross section (crossing blocks 2 and 4) showing the volume fraction of each phase at different times is shown. These results are identical to results of simulations using a one block configuration [GST93].

Figure 2 Convergence history for 3D turbulent flow in a cylindrical pipe.



6 Conclusions

The multiblock solver presented in this paper has been shown to work well in situations involving various block configurations as well as for different flow situations involving both single- and multi-phase flows. We have focused here on some fairly simple examples involving incompressible flows since in that situation pressure corrections are felt instantaneously across the entire domain. The SIMPLE solution methodology is widely used in multi purpose CFD simulators and is potentially applicable to all Mach number regimes ranging from incompressible laminar flows to supersonic flows. The multiblock approach is well suited for implementation on high performance machines.

Acknowledgement

This research was supported by the Research Council of Norway through programme number STP-30074.

REFERENCES

[BW87] Burns A. and Wilkes N. (July 1987) A finite difference method for the computation of fluid flows in complex three dimensional geometries. Technical Report HL87 1327 (07), Harwell Laboratory.

[Dec92] Deconinck H. (ed) (March 1992) *VKI Lecture Series*, Rhode St. Genese, Belgium. von Karman Institute.

[GST93] Gjesdal T., Storvik I., and Teigland R. (April 1993) MUSIC version 1.0, implementation issues and test examples. CMR Report CMR-93-A20001, Christian Michelsen Research AS, Fantoftvegen 38, N-5036 Fantoft, NORWAY.

[Hac94] Hackbusch W. (1994) *Iterative Solution of Large Sparse Systems of Equations*, volume 95 of *Applied Mathematical Sciences.* Springer Verlag.

[KG93] Kuerten H. and Geurts B. (1993) Compressible turbulent flow simulation with a multigrid multiblock method. In Melson N. D., Manteuffel T. A., and McCormick S. F. (eds) *Sixth Copper Mountain Conference on Multigrid Methods*, volume CP 3224, pages 305–315. NASA, Hampton, VA.

[Mel90] Melaaen M. (1990) *Analysis of curvilinear non-orthogonal coordinates for numerical calculation of fluid flow in complex geometries.* Dr. ing. thesis, NTH, Trondheim, Norway.

[PS72] Patankar S. and Spalding D. (1972) A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *Int. J. Heat Mass Transfer* 15: 1787–1806.

[RC83] Rhie C. and Chow W. (November 1983) A numerical study of the turbulent flow past an isolated airfoil with trailing edge separation. *AIAA Journal* 21: 1525–1532.

[SBG96] Smith B., Bjørstad P., and Gropp W. (1996) *Domain Decomposition, Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press.

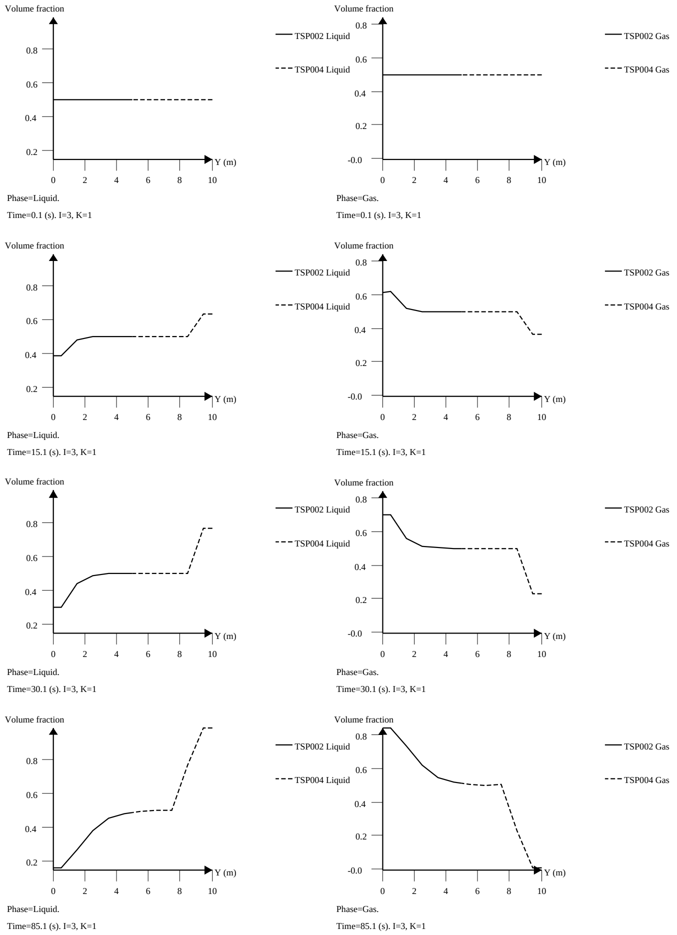
[Tei95] Teigland R. (1995) An additive Schwarz procedure for complex flows. In Thibault P. A. and Bergeron D. M. (eds) *Proc. Third Annual Conference of the CFD Society of Canada*, pages 145–152.

[TF92] Tu J. Y. and Fuchs L. (1992) Overlapping grids and multigrid methods for three dimensional unsteady flow calculations in IC engines. *Int. J. Numer. Meth. Fluids* 15: 693–714.

[VdV92] Van der Vorst H. (1992) Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of non-symmetric linear systems. *SIAM J. Sci. Stat. Comput.* 13: 631–644.

[Wes92] Wesseling P. (1992) *An Introduction to Multigrid Methods.* Pure and Applied Mathematics. Wiley & Sons.

Figure 3 Results for the sedimentation example using 3 SIMPLE iterations per timestep. Volume fraction, left: Liquid phase, right: Gas phase. Solid line block 2, dotted line block 4.



97

Applications of Quasi-Newton Methods for the Numerical Coupling of Some Nonlinear Problems

C.-H. Lai

1 Introduction

In a typical domain decomposition method, substructure solutions are used to construct preconditioners that are to be used in a conjugate gradient algorithm applied to the entire discretised problem. The approach used in this paper is to solve, instead of the entire discretised problem, the reduced interface problem which arisen from domain decomposition methods. This research has been motivated by a current project on viscous/inviscid coupling [LCP96]. In order to couple two models, an iterative scheme first developed by Schwarz for elliptic problems is usually employed [Sch90]. However the rate of convergence is not satisfactory, even for linear problems, without the use of preconditioners.

The interfacial problem along the subproblems interface is usually obtained as certain defects. Early experiences in this context can be found in [Lai93][Lai94]. The approach is based on a two level scheme. At the finer level, each subproblem is described by a nonlinear continuous model and solved independent of other subproblems using a local Newton's method. These subproblem solutions contribute to the evaluation of the defects along the interface of subproblems. At the coarse level, the defect equation is being solved by means of a quasi-Newton method. Two reasons of using quasi-Newton methods. First, the difficulty in computing the Jacobian matrix and second, the analytic form of the Jacobian is not known. A comparison of various quasi-Newton methods for a linear convection-diffusion problem can be found in [Lai94].

This paper is organised as follows. First, a simple interface problem is introduced followed by a description of some quasi-Newton methods. The performance of the nonlinear coupling on distributed computing environment is discussed with numerical examples.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

2 A Simple Interface Problem

We consider the interface problem of the following two-point boundary value problem,

$$\frac{d^2\phi}{dx^2} = g(x, \phi, \frac{d\phi}{dx}), \qquad \in \qquad \Omega = \{x | a < x < b\} \tag{2.1}$$

subject to Dirichlet boundary conditions $\phi(a) = \phi_a$ and $\phi(b) = \phi_b$. The domain Ω is split into $s+1$ nonoverlapped subdomains, Ω_k, $k = 1, 2, \cdots, s+1$, such that $\Omega = \{\cup_{k=1}^{s+1}\Omega_k\} \cup \{\cup_{k=1}^{s}\Gamma_k\}$ where $\Omega_k = \{x | x_{k-1} < x < x_k\}$, $\partial\Omega_k = \{x_{k-1}, x_k\}$, $\Gamma_k = x_k$. Each of the subdomains has the following associated two-point boundary value problem,

$$\frac{d^2 u_k}{dx^2} = g(x, u_k, \frac{du_k}{dx}), \qquad \in \qquad \Omega_k \tag{2.2}$$

subject to boundary conditions $u_k(x_{k-1}) = \lambda_{k-1}$ and $u_k(x_k) = \lambda_k$, and $u_1(x_0) = \phi_a$ and $u_{s+1}(x_{s+1}) = \phi_b$ where $x_0 = a$, $x_{s+1} = b$. Let $u_k = u_k(x; \boldsymbol{\lambda})$ denote the solution of (2.2) in Ω_k, where $\boldsymbol{\lambda} = [\lambda_1 \ \lambda_2 \ \cdots \ \lambda_s] \in \Omega_D \subset R^s$. In order to obtain unique values of $\phi'(x_k)$, $k = 1, 2, \cdots, s$, we define the defect $\mathbf{D} : \Omega_D \subset R^s \to R^s$ as

$$\mathbf{D}(\boldsymbol{\lambda}) = [D_k(\boldsymbol{\lambda})] \equiv [\frac{\partial}{\partial x} u_k(x_k; \boldsymbol{\lambda}) - \frac{\partial}{\partial x} u_{k+1}(x_k; \boldsymbol{\lambda})] \tag{2.3}$$

and require to solve the defect equation $\mathbf{D}(\boldsymbol{\lambda}) = 0$. The continuity of the function ϕ across the interfaces is implicit in (2.2). The defect equation guarantees the continuity of ϕ' across the interfaces. It can be easily seen that $\mathbf{D} \in C^1(\Omega_D)$. In the case of two subdomains, the defect equation has one unknown. In the case of multi-subdomain, the Jacobian matrix $J(\boldsymbol{\lambda}) = \mathbf{D}'(\boldsymbol{\lambda})$ is a nonsymmetric tridiagonal matrix [Lai94]. If $\boldsymbol{\lambda} = \boldsymbol{\lambda}^*$ is a root of $\mathbf{D}(\boldsymbol{\lambda}) = 0$, then the function

$$\phi(x) = \begin{cases} \lambda_{k-1}^* & x = x_{k-1} \\ u_k(x; \boldsymbol{\lambda}^*) & x_{k-1} < x < x_k, \\ \lambda_k^* & x = x_k \end{cases} \qquad k = 1, 2, \cdots, s+1 \tag{2.4}$$

where $\lambda_0^* = \phi_a$ and $\lambda_{s+1}^* = \phi_b$, is a solution of (2.1).

Note that (2.3) is the equilibrium state of the variables $\frac{\partial}{\partial x} u_k(x_k; \boldsymbol{\lambda})$ and $\frac{\partial}{\partial x} u_{k+1}(x_k; \boldsymbol{\lambda})$. There are other equilibrium states, e.g. integrate the difference of the derivatives along the interface, or perhaps other physically viable states. Once the mathematical interface coupling is defined, the defect equation can be easily set up and contributions to the defect equation from different subdomains can be separately computed which ensures distributed computing tasks.

3 Some Quasi-Newton Methods

In this section we consider a number of quasi-Newton methods for the solution of the defect equation $\mathbf{D}(\boldsymbol{\lambda}) = 0$. Let $\mathbf{D} : \Omega_D \subset R^s \to R^s$ where Ω_D is an open and convex set, $\mathbf{D} \in C^1(\Omega_D)$, $\mathbf{D}(\boldsymbol{\lambda}^*) = 0$, $J(\boldsymbol{\lambda}^*)$ nonsingular, and, for all $\boldsymbol{\lambda} \in \Omega_D$

$$||J(\boldsymbol{\lambda}) - J(\boldsymbol{\lambda}^*)|| \leq L||\boldsymbol{\lambda} - \boldsymbol{\lambda}^*||^p \tag{3.5}$$

for some norm $||.||$, and some L, $p > 0$. The general quasi-Newton method for the solution of $\mathbf{D}(\boldsymbol{\lambda}) = 0$ is given by

$$\boldsymbol{\lambda}^{(n+1)} = \boldsymbol{\lambda}^{(n)} - \alpha_n^{-1}\mathbf{D}(\boldsymbol{\lambda}^{(n)}) \tag{3.6}$$

where α_n is a nonsingular matrix approximating the Jacobian matrix J($\boldsymbol{\lambda}$). The convergence result of such an algorithm can be found in [DM74]. We choose six algorithms for the present comparison: a modified Newton method, Broyden's method, Schubert's method, Schubert-Kim method, Schubert-Powell method and an adaptive-α method. We are interested to apply these methods to the linear system $\mathbf{D}(\boldsymbol{\lambda}) \equiv J\boldsymbol{\lambda} - \mathbf{b} = 0$ where J is an $s \times s$ matrix as well as to the nonlinear system $\mathbf{D}(\boldsymbol{\lambda}) = 0$ in general. For some interface problems, the sparsity structure of the Jacobian matrices is known, therefore one should employ the so-called Schubert's update rather than Broyden's update. The difference between Broyden's method and Schubert's method is that the former does not take care of the sparsity of the Jacobian matrix while the latter preserves the sparseness structure of the Jacobian matrix. The naming convention for the algorithms being tested in this paper is that any name beginning with Schubert has its sparseness structure of the Jacobian matrix being preserved.

In order to avoid the update of the approximate Jacobian every iteration, a modified Newton method is applied here in such a way that the Jacobian is only calculated once and is used in all subsequent iteration.

Algorithm 3.1 Modified Newton's Method [Lai94]. Given $\boldsymbol{\lambda}^{(0)}$ and α_0, compute $\boldsymbol{\lambda}^{(1)}$ using (3.6). Then evaluate $J(\boldsymbol{\lambda}^{(1)})$ by means of a finite difference approximation. Finally use (3.6) to compute $\boldsymbol{\lambda}^{(n+1)}$ by choosing $\alpha_n = J(\boldsymbol{\lambda}^{(1)})$, $n = 1, 2, \ldots$.

α_0 is chosen as a diagonal matrix or in such a way that its sparseness is the same as that of the Jacobian matrix. Also α_n is calculated once and is kept for all subsequent iteration because it is expensive to evaluate the Jacobian matrix in every iterative step.

One classical technique of choosing α_n is called Broyden's update. Let $Q_{u,v} = \{\hat{\alpha} \in \mathbf{L}(R^s), \mathbf{u}, \mathbf{v} \in R^s : \hat{\alpha}\mathbf{u} = \mathbf{v}\}$. Then Broyden's update is obtained as the solution to the minimisation problem [DS79] $\min\{||\hat{\alpha} - \alpha_n||_F : \hat{\alpha} \in Q_{\mathbf{s}_n,\mathbf{y}_n}\}$ where $\mathbf{s}_n = \boldsymbol{\lambda}^{(n+1)} - \boldsymbol{\lambda}^{(n)}$, $\mathbf{y}_n = \mathbf{D}(\boldsymbol{\lambda}^{(n+1)}) - \mathbf{D}(\boldsymbol{\lambda}^{(n)})$, $||.||_F$ is the Frobenius norm. The solution of the minimisation problem is given by

$$\alpha_{n+1} = \alpha_n + \frac{\mathbf{D}(\boldsymbol{\lambda}^{(n+1)})\mathbf{s}_n^T}{< \mathbf{s}_n, \mathbf{s}_n >} \tag{3.7}$$

Suppose $\hat{W}$ and W are nonsingular matrices in $\mathbf{L}(R^s)$, then a weighted update can also be obtained as the solution to the minimisation problem [DS79] $\min\{||\hat{W}(\hat{\alpha} - \alpha_n)W||_F : \hat{\alpha} \in Q_{\mathbf{s}_n,\mathbf{y}_n}\}$ and is given by

$$\alpha_{n+1} = \alpha_n + \frac{\mathbf{D}(\boldsymbol{\lambda}^{(n+1)})\mathbf{v}_n^T}{< \mathbf{v}_n, \mathbf{s}_n >} \tag{3.8}$$

where $\mathbf{v}_n = W^{-T}W^{-1}\mathbf{s}_n$.

Algorithm 3.2 Broyden's Method [DS79]. Given $\boldsymbol{\lambda}^{(0)}$ and α_0, compute $\boldsymbol{\lambda}^{(n+1)}$ using (3.6) and α_{n+1} using (3.7), for $n = 0, 1, \ldots$.

For simplicity α_0 is chosen as a diagonal matrix. However the subsequent generated matrices, α_n, are full matrices. The inverses of these matrices are expensive and there is no guarantee that these matrices are nonsingular.

Algorithm 3.3 Schubert's Method [Mar79]. Perform the same steps as that given in Algorithm 3.2 but the successive updates α_{n+1} are made in such a way that the sparseness structure is preserved.

For simplicity α_0 is chosen as a diagonal matrix. As in the previous case, the sequence of matrices generated by (3.6) is not necessarily nonsingular.

Kim and Tewarson [KT92] proposed a weighted mean of (3.7) and (3.8) with $\mathbf{v}_n = -\alpha_n^T \mathbf{D}(\boldsymbol{\lambda}^{(n)})$, i.e.

$$\alpha_{n+1} = \alpha_n + (1-\mu)\frac{\mathbf{D}(\boldsymbol{\lambda}^{(n+1)})\mathbf{s}_n^T}{< \mathbf{s}_n, \mathbf{s}_n >} + \mu \frac{\mathbf{D}(\boldsymbol{\lambda}^{(n+1)})\mathbf{v}_n^T}{< \mathbf{v}_n, \mathbf{s}_n >} \tag{3.9}$$

where μ is chosen to satisfy $\|\frac{\mathbf{D}(\boldsymbol{\lambda}^{(n+1)})\mathbf{s}_n^T}{<\mathbf{s}_n,\mathbf{s}_n>}\|_F = \|\mu\frac{\mathbf{D}(\boldsymbol{\lambda}^{(n+1)})\mathbf{v}_n^T}{<\mathbf{v}_n,\mathbf{s}_n>}\|_F$ from which μ is obtained as

$$\mu = \frac{< -\alpha_n^T \mathbf{D}(\boldsymbol{\lambda}^{(n+1)}), \mathbf{s}_n >^2}{< \mathbf{s}_n, \mathbf{s}_n >< -\alpha_n^T \mathbf{D}(\boldsymbol{\lambda}^{(n+1)}), -\alpha_n^T \mathbf{D}(\boldsymbol{\lambda}^{(n+1)}) >} \tag{3.10}$$

We implemented the weighted update of Kim and Tewarson by means of Schubert's approach, i.e. the sparsity of the Jacobian matrix is preserved.

Algorithm 3.4 Schubert - Kim Method. Given $\boldsymbol{\lambda}^{(0)}$ and α_0, compute $\boldsymbol{\lambda}^{(n+1)}$ using (3.6) and α_{n+1} using (3.9), for $n = 0, 1, \ldots$.

It is worth to note that when $\mu = 1$, the method actually fails to converge for some interface problems, in particular the nonlinear boundary value problem described later.

None of the methods described so far has equipped with a technique to advoid singular matrix α_{n+1}. Using the well known determinant property $\det(I + \mathbf{u}\mathbf{v}^T) = 1+ < \mathbf{u}, \mathbf{v} >$ for any $\mathbf{u}$ and $\mathbf{v}$ in R^s, one can deduce that if α_n is nonsingular then, α_{n+1} is nonsingular if and only if $< \mathbf{s}_n, \alpha_n^{-1}\mathbf{y}_n > \neq 0$. Powell defined a modification to Broyden's method given by [MT76]

$$\alpha_{n+1} = \alpha_n + \theta_n \frac{\mathbf{D}(\boldsymbol{\lambda}^{(n+1)})\mathbf{s}_n^T}{< \mathbf{s}_n, \mathbf{s}_n >} \tag{3.11}$$

where θ_n is chosen so that α_{n+1} is nonsingular. In other words, we require

$$|\det \alpha_{n+1}| \geq \eta |\det \alpha_n|, \qquad |\theta_n - 1| \leq \eta \tag{3.12}$$

for any given $\eta \in (0, 1)$. Using the above determinant property, one can easily deduce the following relation

$$\theta_n = \begin{cases} 1, & |\gamma_n| \geq \eta \\ \frac{1-\text{sign}(\gamma_n)\eta}{1-\gamma_n}, & |\gamma_n| < \eta \end{cases} \tag{3.13}$$

where $\gamma_n =< \mathbf{s}_n, \alpha_n^{-1}\mathbf{y}_n > / < \mathbf{s}_n, \mathbf{s}_n >$, and sign $(0) = 1$. We implement, in Algorithm 3.5, the above modification by means of Schubert's approach.

Algorithm 3.5 Schubert - Powell Method [Mar79]. Given $\boldsymbol{\lambda}^{(0)}$ and α_0, compute $\boldsymbol{\lambda}^{(n+1)}$ using (3.6) and α_{n+1} using (3.11) and (3.13) for $n = 0, 1, \ldots$.

It should be noted that while Powell's modification to Broyden's method leads to global and superlinear convergence in the case of linear systems, it does not hold for general nonlinear functions [MT76].

Finally, we describe an algorithm based on a sequence of adaptive parameters [Lai94]. Here the technique for a scalar defect equation is essentially an optimal one-point iteration method where α_n is obtained by setting $\mathbf{G}' = 0$, where $\mathbf{G} = \boldsymbol{\lambda}^{(n)} - \alpha_n^{-1}\mathbf{D}(\boldsymbol{\lambda}^{(n)})$. This adaptive parameter α_n is equivalent to the scalar ϵ-algorithm [Lai94]. An adaptive α for the extension [Lai94] to s-dimensional problems is

$$\alpha_{n+1} = \alpha_n \frac{\|\mathbf{D}(\boldsymbol{\lambda}^{(n+1)}) - \mathbf{D}(\boldsymbol{\lambda}^{(n)})\|}{\|\mathbf{D}(\boldsymbol{\lambda}^{(n)})\|} \tag{3.14}$$

Algorithm 3.6 Adaptive α [Lai94]. Given $\boldsymbol{\lambda}^{(0)}$ and α_0, compute $\boldsymbol{\lambda}^{(n+1)}$ using (3.6) and α_{n+1} using (3.14), for $n = 0, 1, \ldots$.

Since an arbitrary initial approximation is chosen, there is no guarantee that $\boldsymbol{\lambda}^{(0)}$ is sufficiently close to $\boldsymbol{\lambda}^*$, in particular for nonlinear problems, which is an essential requirement for global convergence. One way to generate better initial approximation for nonlinear problems is to use Algorithm 3.6 which provides a small step during the initial few updates of the Jacobian. In the numerical tests shown later, Algorithm 3.6 is employed 3 or 4 times before the other algorithms are employed. The number of iterative updates, n_{it}, as presented in the Tables as shown later includes the above number of initial iterates. The reason for including these initial iterates in the iteration count becomes clear when the feasibility for parallel implementation is discussed. A stopping criterion for the above algorithms is $\|\boldsymbol{\lambda}^{(n)} - \boldsymbol{\lambda}^*\| < \epsilon$ where $\boldsymbol{\lambda}^*$ is given and ϵ is a small tolerance.

4 Performance Analysis

Now the approach involved in the present study is to divide the problem into two levels. At the fine level, the problem is divided into a number of subproblems to be computed in parallel. At the coarse level, the problem is small enough not to warranty for parallel implementation and the computational work is taken as the sequential overhead.

Let M denote the total number of nodes in the entire computational domain. One work unit is defined as the computational work required to solve the discretised problem with M nodes. For steady linear problems, solving the entire computational problem requires one work unit. For nonlinear problems, solving the entire computational problem requires n_s work unit, where n_s is the number of linearisations. For time dependent problems, solving the entire computational problem requires $n_t n_s$ work units where n_t is the number of time steps. Let the entire computational domain be divided into $s + 1$ subdomains, M_k be the number of nodes in the k-th subdomain, $k = 1, 2, \cdots, s+1$ and n_{it} be the number of updates in order to obtain a converged solution $\boldsymbol{\lambda}^{(n_{it})}$ along the interfaces by using a quasi-Newton scheme. Suppose there is a set of $s + 1$ concurrent processors and that the connectivity is the same as the layout of the subdomains. Since most of the computational work is devoted to the solutions of subproblems, it is possible to estimate the parallel computing time by taking the sum of the maximum work unit in any iteration involved in the subproblem solutions, i.e.

$$\tau = \sum_{n=1}^{n_{it}} \max_{1 \leq k \leq s+1} \{\frac{M_k}{M}\} \tag{4.15}$$

Hence a quasi-Newton algorithm described previously is considered as a feasible parallel algorithm provided $\tau < n_s$. Note that the present performance monitor does not include the overheads required to solve the linear systems and that such overhead becomes negligible for Algorithm 3.6.

5 Numerical Examples

We consider the following convection dominant flow problem,

$$\frac{d^2\phi}{dx^2} - \gamma\frac{d\phi}{dx} = 0, \qquad \phi(0) = 0, \phi(1) = 1 \tag{5.16}$$

where $\gamma \gg 1$. The domain is subdivided into $s+1$ subdomains with interfaces, Γ_k, $k = 1, 2, \ldots, s$, distributed evenly across the domain. We use exact solutions in each of the subdomains and are interested to compare the number of iterations, n_{it}, required to update the function values along the interfaces. The value ϵ of the stopping criterion is chosen to be 0.5×10^{-5}. For the present studies, the defect equation system for the convection diffusion problem is a linear system and Algorithm 3.5 is included in the test set. The number of iterations are presented in the middle column of Table 1 and values obtained by using (4.15) are presented in the third column of Table 1. Note that an "x" represents divergence of the test. More results can be found in ([Lai93]).

Table 1 Linear problem: $\gamma = 50$.

			$s+1$					$s+1$		
	4	8	16	32	64	4	8	16	32	64
Alg.			n_{it}					τ		
3.1	2	2	2	2	2	0.500	0.250	0.125	0.063	0.031
3.2	6	14	25	48	88	1.500	1.750	1.563	1.500	1.375
3.3	7	13	19	21	36	1.750	1.625	1.188	0.656	0.563
3.4	7	13	18	24	x	1.750	1.625	1.125	0.750	x
3.5	9	14	16	22	36	2.250	1.750	1.000	0.688	0.563
3.6	10	19	31	51	102	2.500	2.375	1.938	1.594	1.594

Obviously, Algorithm 3.1 is very attractive but the construction of $J(\boldsymbol{\lambda}^{(1)})$ requires $2s$ subproblem solvers. Algorithms 3.3, 3.4, and 3.5 are more efficient compare with 3.2 and 3.6.

Table 2 Nonlinear problem: $a = 1$ and $b = 0.5$, 161 mesh points.

	$s+1$				$s+1$			
($n_s = 4$)	4	8	16	32	4	8	16	32
Alg.		n_{it}				τ		
3.1	12	11	12	11	3.000	1.375	0.750	0.343
3.2	11	17	34	57	2.750	2.125	2.125	1.781
3.3	10	18	19	17	2.500	2.250	1.188	0.531
3.4	11	17	x	x	2.750	2.125	x	x
3.6	12	20	72	114	3.000	2.500	4.500	3.563

Next, we consider the nonlinear elliptic boundary value problem,

$$\frac{d}{dx}(K(\phi)\frac{d\phi}{dx}) = f(x), \qquad \phi(0) = 0, \phi(1) = 1, \tag{5.17}$$

where $K(\phi) = a + b\phi$ is the thermal conductivity and $f(x) = 2a + 6bx^2$ for real numbers of a and b. The analytic solution of the problem is $\phi = x^2$. Since the problem is nonlinear, therefore the defect equation is also nonlinear. In order to solve the nonlinear subproblem, a linearisation based on Newton's method is applied at the subproblem level and a finite difference method is then applied to the linearised subproblem, i.e. $F'(u_k)(u_k^{new} - u_k) = -F(u_k)$ where $F(u_k) = f(x) - \frac{d}{dx}(K(u_k)\frac{du_k}{dx})$. The resulting set of linear equation is solved by a Gaussian elimination. It is not intended in this paper to study the efficiency of linear solvers at the subproblem level, and there are plenty of fast linear solvers available. The important issue here is the convergence rate of the quasi-Newton method, i.e. the number of updates, n_{it}, of the defect equation along the interface. Results for $a = 1.0$ and $b = 0.5$ are presented for the cases of 161 and 321 mesh points. Since the defect equation is a nonlinear system, Algorithm 3.5 does not guarantee that the approximate Jacobian matrices to be nonsingular and therefore it is not included in the test set. Tables 2 and 3 show the relationship between n_{it} and $s + 1$ for the nonlinear boundary value problem.

Table 3 Nonlinear problem: $a = 1$ and $b = 0.5$, 321 mesh points.

	$s+1$					$s+1$				
($n_s = 4$)	4	8	16	32	64	4	8	16	32	64
Alg.			n_{it}					τ		
3.1	13	11	13	14	30	3.250	1.375	0.813	0.438	0.469
3.2	11	17	36	57	121	2.750	2.125	2.250	1.781	1.891
3.3	10	19	19	20	21	2.500	2.375	1.188	0.625	0.328
3.4	11	18	28	x	x	2.750	2.250	1.750	x	x
3.6	13	23	61	126	206	3.250	2.875	3.813	3.938	3.219

For test problems with 161 mesh points, one cannot have 64 subdomains. The weighted mean method described in Algorithm 3.4 seems to be less attractive for nonlinear problems because it diverges for a fairly small number of interfacial unknowns. In general, quasi-Newton methods performs better for nonlinear boundary value problems compare with linear boundary value problems.

6 Conclusions

A novel application of quasi-Newton methods for the solution of interface problems arisen from domain decomposition methods is examined. Performance analysis shows that Algorithms 3.2, 3.3, and 3.4 have been identified to be suitable for nonlinear problems. Algorithm 3.6 is included as a feasible parallel algorithm for nonlinear problems but is not suitable for linear problems. Algorithm 3.1 is extremely effective for linear problems only if enough parallel processors are available to construct the Jacobian. Algorithm 3.6 is also important in the sense that it provides stable early iterates in Algorithm 3.1 at a cheap overhead in order to produce an approximate solution in the neighbour of the exact solution. Extension to 2-D problems is currently being studied.

REFERENCES

[DM74] Dennis J. J. and Moré J. (1974) A characterization of superlinear convergence and its application to quasi-newton methods. *Math Comp* 28: 549–560.

[DS79] Dennis J. J. and Schnabel R. (1979) Least change secant updates for quasi-newton methods. *SIAM Review* 21: 443–459.

[KT92] Kim S. and Tewarson R. (1992) A quasi-newton method for solving nonlinear algebraic equations. *Computers Math. Applic.* 24: 93 – 97.

[Lai93] Lai C.-H. (1993) Comparing quasi-newton method for solving sparse interface problem. CWI Technical Report NM-R9303, CWI, Amsterdam, the Netherlands.

[Lai94] Lai C.-H. (1994) An iteration scheme for non-symmetric interface operator. In et. al. R. G. (ed) *Contemporary Mathematics*, number 157, pages 279–285. American Mathematical Society.

[LCP96] Lai C.-H., Cuffe A., and Pericleous K. (1996) A domain decomposition algorithm for viscous/inviscid coupling. *Advances in Engineering Software* 26: 151–159.

[Mar79] Marwil E. (1979) Convergence results for schubert's method for solving sparse nonlinear equations. *SIAM J. Num. Anal.* 16: 588–604.

[MT76] Moré J. and Trangenstein J. (1976) On the global convergence of broyden's method. *Math. Comp.* 30: 523–540.

[Sch90] Schwarz H. (1890) über einen grenzübergang durch alternierendes verfahren. *Gesammelte Mathematische Abhandlungen* 41: 133–143.

98

A Characteristic Domain Decomposition Algorithm for Two-Phase Flows with Interfaces

Hong Wang and Brit Gunn Ersland

1 Introduction

The mathematical model that describes the process of an immiscible displacement of oil by water in reservoir production or other two-phase fluid flows in porous media leads to a strongly coupled system of a degenerated nonlinear advection-diffusion equation for saturation and an elliptic equation for pressure and velocity. The hyperbolic nature, strong coupling, and nonlinearity of the system and the degeneracy of the diffusion makes numerical simulation a challenging task. Many numerical methods suffer from serious non-physical oscillations, excessive numerical dispersion, and/or a combination of both [CJ86, Ewi84]. Previously, Espedal, Ewing, and coworkers developed a characteristic, operator-splitting technique in effectively solving two-phase fluid flow problems [DEES90, EE87]. In practice, a reservoir often consists of different subdomains with different porosities and permeabilities. In the case of single-phase fluid flows the concentration and total flux are continuous across the interfaces between different subdomains since the diffusion never vanishes. Our earlier work addressed numerical simulation to linear transport equations arising in single-phase flows with interfaces [WDE+94]. However, in the case of two-phase fluid flows the saturation equation itself is nonlinear and different subdomains have different capillary pressure curves. The continuity of capillary pressure across interfaces implies a jump discontinuity of the water saturation at the same locations. The jump discontinuity of the saturation at the interfaces might incur some oscillations around the interfaces, which can be propagated into the domain and destroy the overall accuracy. Hence, great care has to be taken in the development of an effective solution procedure for the simulation of two-phase fluid flows in porous media with interfaces.

This paper describes a characteristic-based, non-overlapping domain decomposition algorithm for solving the saturation equation in two-phase fluid flows with interfaces. First, with the known saturation at the previous time step one obtains an approximate Dirichlet boundary condition at the outflow domain interface by integrating the

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

saturation equation (ignoring the capillary pressure term) along characteristics. With the approximate outflow Dirichlet boundary condition at the domain interface and the given boundary condition at the physical inflow boundary one can close the system on the current subdomain and applies the characteristic operator-splitting procedure [DEES90, EE87] to solve the full saturation equation (including the capillary pressure effect). Second, one uses the continuity of capillary pressure across the domain interface to pass the value of saturation as an approximate inflow Dirichlet boundary condition to the next subdomain, one then applies the same characteristic operator-splitting procedure to solve the saturation equation on the current subdomain. Third, according to the overall loss or gain of mass one adjusts the approximate outflow Dirichlet boundary condition at the domain interface to iterate between different subdomains until the algorithm converges. Finally, a mixed method is adopted to solve the pressure equation due to its accurate approximation to the velocity field and its local mass conservation property.

The rest of the paper is organized as follows: In Sections 2 and 3 we formulate the problem and discuss related solution techniques. In Section 4 we present a domain decomposition algorithm for the two-phase fluid flow problems with interfaces. In Section 5 we present some numerical results to show the promise of the method.

2 Problem Formulation

A suitable mathematical model for the total Darcy velocity $\underline{u}$, the total pressure p, and the water saturation $S \in [0,1]$ in an incompressible displacement of oil by water in a porous medium can be described by the following set of partial differential equations [CJ86]:

$$\begin{array}{rll} \nabla \cdot \underline{u}(\underline{x},t) & = q_1(\underline{x},t), & (\underline{x},t) \in \Omega \times [0,T], \\ \underline{u}(\underline{x},t) & = -\underline{K}(\underline{x})(\lambda_o(S)+\lambda_w(S))\nabla p(\underline{x},t), & (\underline{x},t) \in \Omega \times [0,T], \\ \underline{u}(\underline{x},t)\cdot \underline{n}(\underline{x}) & = q_2(\underline{x},t), & (\underline{x},t) \in \partial\Omega \times [0,T], \end{array} \tag{1}$$

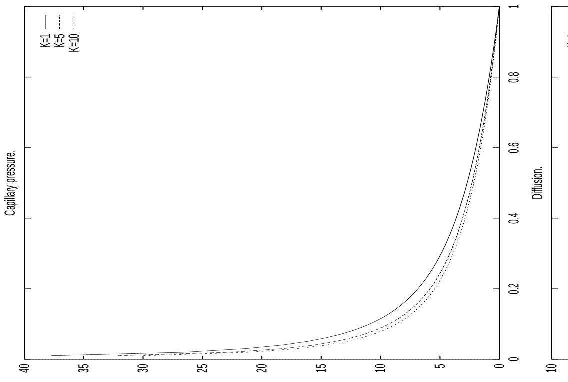
and

$$\begin{array}{rll} \phi(\underline{x})\dfrac{\partial S}{\partial t} + \nabla \cdot (f(S)\underline{u} - \varepsilon \underline{D}(S,\underline{x})\nabla S) & = q_3(\underline{x},t), & (\underline{x},t) \in \Omega \times [0,T], \\ (f(S)\underline{u} - \varepsilon \underline{D}(S,\underline{x})\nabla S)\cdot \underline{n}(\underline{x}) & = q_4(\underline{x},t), & (\underline{x},t) \in \partial\Omega \times [0,T], \\ S(\underline{x},0) & = S_0(\underline{x}), & \underline{x} \in \Omega, \end{array} \tag{2}$$

where Ω is the physical domain, $\underline{K}(\underline{x})$ is the absolute permeability tensor of the medium, λ_i, $i = o, w$, denotes the water and oil mobilities respectively, $q_1(\underline{x},t)$ and $q_3(\underline{x},t)$ are source terms, $q_2(\underline{x},t)$ and $q_4(\underline{x},t)$ are the prescribed boundary conditions, $\underline{n}(\underline{x})$ is the unit outward normal vector, $\varepsilon << 1$ is a scaling factor to the diffusion term, p_c is the capillary pressure, and $f(S)$ and $\underline{D}(S,\underline{x})$ are the fractional flow function and diffusion term given by

$$\begin{array}{ll} f(S) & = \dfrac{\lambda_w(S)}{\lambda_w(S)+\lambda_o(S)}, \\ \underline{D}(S,\underline{x}) & = \underline{K}(\underline{x})\dfrac{\lambda_w(S)\lambda_o(S)}{\lambda_o(S)+\lambda_w(S)}\dfrac{dp_c}{dS}. \end{array} \tag{3}$$

Figure 1 Capillary pressure and diffusion as functions of saturation S for permeabilities 1, 5, and 10.



Note that the two equations in (1) form a second-order elliptic equation for the pressure $p(\underline{x}, t)$ and are coupled to the saturation equation (2) through the saturation S in the coefficients. On the other hand, saturation equation (2) is a nonlinear advection-diffusion equation and is coupled to the pressure equation (1) through the Darcy velocity $\underline{u}$. Furthermore, in the mathematical model the diffusion term $\underline{D}(S, \underline{x})$ vanishes at $S = 0$ and $S = 1$, which is an idealized case since physically $\underline{D}(S, \underline{x})$ vanishes for $S \in [0, S^{ir}]$ or $S \in [1 - S^{ir}, 1]$ with S^{ir} being the irreducible saturation value. The fractional flow function $f(S)$ defined in (3) is typically an S-shaped curve of saturation S and degenerates at $S = 0$ (with the same understanding). Because the saturation profile is usually a decreasing function in space, as time evolves $f(S)$ tends to force a shock discontinuity to develop in S while the diffusion term $\underline{D}(S, \underline{x})$ tends to prevent the shock from forming. The dynamic process could be fairly complicated because the diffusion degenerates in front of the steep saturation front. It depends on the interaction between the advection and diffusion terms, in particular, on the rates at which $\underline{D}(S, \underline{x})$ and $f(S)$ tend to zero as S tends to zero.

When the physical domain Ω is composed of different media, the different porosities and permeabilities result in different capillary pressure curves on each subdomain (Figure 1). Across an interface Γ the phase pressures are continuous and mass is conserved, leading to the following interface conditions

$$\begin{aligned} p_c(S)|_{\Gamma_-} &= p_c(S)|_{\Gamma_+}, \\ \underline{u} \cdot \underline{n}|_{\Gamma_-} &= \underline{u} \cdot \underline{n}|_{\Gamma_+}, \\ (f(S)\underline{u} - \varepsilon \underline{D}(S, \underline{x}) \nabla S) \cdot \underline{n}|_{\Gamma_-} &= (f(S)\underline{u} - \varepsilon \underline{D}(S, \underline{x}) \nabla S) \cdot \underline{n}|_{\Gamma_+}. \end{aligned} \tag{4}$$

The continuity of capillary pressure p_c across an interface Γ implies the discontinuity of the saturation across the interface (Figure 1). One has to resolve the discontinuity carefully so that no spurious effects will be propagated into the domain.

3 Operator Splitting Techniques

Extensive research has been carried out for the numerical simulation of system (1)–(2) without interfaces. Various techniques have been developed to decouple and linearize the system, including a fully coupled and fully implicit linearization strategy, an IMPES (IMplicitly advances the Pressure and Explicitly updates the Saturation in time) strategy, and a sequential time stepping strategy [Ewi84]. Different numerical methods, including the standard Galerkin finite element method, the cell-centered finite difference method, the finite volume method, and the mixed finite element method, have been used to solve the pressure equation [CJ86, DEES90, DEW83, EE87, Ewi84]. We used the mixed method to solve the pressure equation due to its accurate approximation to the velocity field and its local mass conservation property. Because the normal component of the velocity field is continuous, the discrete algebraic system for the pressure equation is in fact the same as that with no interfaces. Hence, one can solve the global system as usual. Alternatively, one can use a domain decomposition procedure to solve the pressure equation on each subdomain iteratively. We refer the interested readers to [BW86, SBG96] and the references therein for details.

For simplicity of exposition we consider a one-dimensional analogue of equation (2). Notice that equation (2) is almost hyperbolic due to the small parameter $\varepsilon << 1$. An effective solution procedure for solving the dominating advective part of equation (2)

$$\phi(x)\frac{\partial S}{\partial t} + \frac{\partial}{\partial x}(f(S)u) = 0 \tag{5}$$

is to discretize equation (5) along the characteristics, which allows large time steps to be used in the numerical simulation. Because equation (5) may have more than one solution due to the shape of the fractional flow function $f(S)$, one cannot directly apply the modified method of characteristics [DR82] to equation (5). We follow the work of Espedal, Ewing, and coworkers [DEES90, EE87] and split the fractional flow function $f(S)$ into two parts by

$$f(S) = \bar{f}(S) + b(S)S, \tag{6}$$

with

$$\bar{f}(S) = \begin{cases} \dfrac{f(S_{BL})}{S_{BL}}S, & \text{if } 0 \le S \le S_{BL}, \\ f(S), & \text{if } S_{BL} < S \le 1. \end{cases} \tag{7}$$

Here the Buckley-Leverett shock saturation S_{BL} is defined by

$$f^{'}(S_{BL}) = \frac{f(S_{BL})}{S_{BL}}. \tag{8}$$

Because $\bar{f}(S)u$ gives the unique physical velocity for an established shock, we use this operator splitting and rewrite equation (2) along the characteristics as

$$\psi(x)\frac{\partial \bar{S}^{n+1}}{\partial \tau} \equiv \phi(x)\frac{\partial \bar{S}^{n+1}}{\partial t} + \bar{f}^{'}(\bar{S}^{n+1})u\frac{\partial \bar{S}^{n+1}}{\partial x} = 0, \tag{9}$$

and

$$\psi(x)\frac{\partial S^{n+1}}{\partial \tau} + \frac{\partial}{\partial x}\Big(b(\bar{S}^{n+1})S^{n+1}u - \varepsilon D(\bar{S}^{n+1}, x)\frac{\partial S^{n+1}}{\partial x}\Big) = q_3(x, t^{n+1}). \tag{10}$$

From the definition of $\bar{f}$ it follows that the characteristic direction is uniquely determined by equation (9) since the shape of $\bar{f}$ allows only a rarefaction wave and a contact discontinuity for a non-increasing saturation profile. Thus, the hyperbolic equation (9) is discretized by integrating backwards along the characteristics

$$x^* = x - \bar{f}'(S^{n*})\Delta t, \tag{11}$$

where $S^{n*} = S(x^*, t^n)$ and $\Delta t = t^{n+1} - t^n$ is the time step.

Note that the characteristics determined by equation (9) are all straight lines in the (x, t) plane. If equation (9) is solved exactly, the only change in the solution along the characteristics is due to diffusion (and possibly the source term which vanishes except at wells). Thus, we solve equation (10) by the modified method of characteristics [DEES90, DR82, EE87]

$$\begin{aligned} &\int_\Omega \phi \frac{S^{n+1} - S^{n*}}{\Delta t} w d\Omega + \int_\Omega \Big(\varepsilon D(S^{n*}, x) \frac{\partial S^{n+1}}{\partial x} - b(S^{n*}) S^{n+1} u\Big) \frac{\partial w(x)}{\partial x} d\Omega \\ &= \int_\Omega q_3 w d\Omega, \forall w(x) \in H_0^1(\Omega). \end{aligned} \tag{12}$$

Here a characteristic tracking is used for the advection term, and the quadratic Petrov-Galerkin method is used for the diffusion term and the residual advection term where the trial functions S are chosen to be hat functions and the test functions $w(x)$ are constructed by adding an quadratic perturbation to the hat functions [DEES90, EE87].

4 A Characteristic Domain Decomposition Algorithm for System (1)–(2) with Interfaces

We now describe a characteristic domain decomposition algorithm for solving the system (1)–(2) with interfaces. We adopt a sequential solution strategy to decouple and linearize the system [DEES90, DEW83]. For the domain decomposition techniques for pressure equations with interfaces we refer the interested readers to [BW86, SBG96] for details. We present the algorithm for a one-dimensional problem on $\Omega = (a, b)$ with one interface at $a < d < b$. Let N be a positive integer, $\Delta t = T/N$, and $t^n = n\Delta t$.

Initialization

Substitute the initial condition $S(x, 0)$ for S in (1) and solve equations (1) at t^0 by the mixed method to obtain the Darcy velocity $u^0(x)$, where $u^n(x) = u(x, t^n)$.

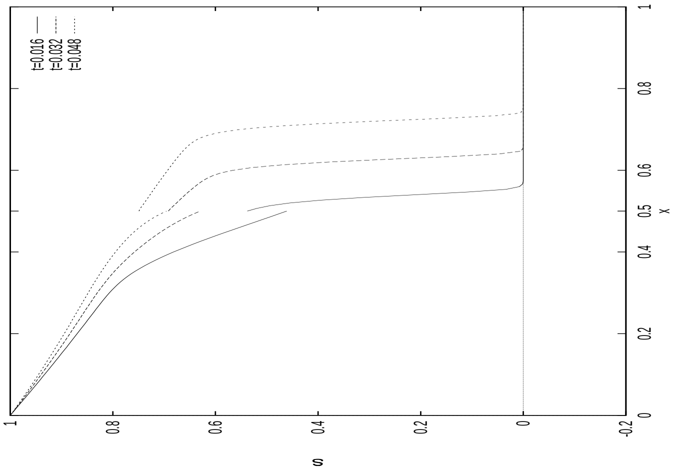
```
for $n = 0, 1, \ldots, N-1$ do
  for $l = 0, 1, \ldots, l_M - 1$ do
  L1. For $l = 0$, in equation (2) approximate $u^{n+1}(x)$ by $u_0^{n+1}(x) = u^n(x)$ or
      $2u^n(x) - u^{n-1}(x)$. For $l \geq 1$, substitute $S_{l-1,k_M}^{n+1}$ for $S$ in (1) and solve
      equations (1) at $t^{n+1}$ by the mixed method to obtain the Darcy velocity
      $u_l^{n+1}$.
```

```
    L2. For $l = 0$, assign $S_{0,0}^{n+1}(d_-) = S^n(d^*)$, where $S_{l,k}^{n+1}(d_-) = \lim_{x \to d, x<d} S_{l,k}(x, t^{n+1})$ and $d^*$ is defined in equation (11) with $x$ being replaced by $d$. For $l \geq 1$, assign $S_{l,0}^{n+1}(d_-) = S_{l-1,k_M}^{n+1}(d_-)$.
    L3. Use the interface condition $p_c^L(S_{l,0}^{n+1}(d_-)) = p_c^R(S_{l,0}^{n+1}(d_+))$ to evaluate $S_{l,0}^{n+1}(d_+)$, where $S_{l,k}^{n+1}(d_+) = \lim_{x \to d, x>d} S_{l,k}^{n+1}(x)$.
    for $k = 0, 1, \ldots, k_M - 1$ do
        if ERROR > TOLERANCE then
            K1. With the given inflow boundary condition at $x = a$ and $S_{l,k}^{n+1}(d_-)$ as the outflow Dirichlet boundary condition at $x = d$, solve equation (12) on the subdomain $(a, d)$ for $S_{l,k}^{n+1}$ at time $t^{n+1}$.
            K2. With $S_{l,k}^{n+1}(d_+)$ as the inflow Dirichlet boundary condition at $x = d$ and the given outflow boundary condition at $x = b$, solve equation (12) on $(d, b)$ for $S_{l,k}^{n+1}$ at $t^{n+1}$ in parallel to the previous step.
            K3. Calculate the mass error $M_{l,k}^{n+1} = \Delta M - \int_\Omega (S_{l,k}^{n+1} - S^n) d\Omega$, where $\Delta M$ is the mass injected at the inflow boundary and through the wells during the time period $[t^n, t^{n+1}]$.
            K4. Update the Dirichlet boundary condition at the interface $x = d$ by $S_{l,k}^{n+1}(d_-) := S_{l,k}^{n+1}(d_-) + \kappa M_{l,k}^{n+1}$, where $\kappa$ is a relaxation parameter.
            K5. Use the interface condition $p_c^L(S_{l,k}^{n+1}(d_-)) = p_c^R(S_{l,k}^{n+1}(d_+))$ to evaluate $S_{l,k}^{n+1}(d_+)$.
        else
            $k = k_M$ and $l = l_M$
        endif
    end
    $k = k_M$
  end
  $l = l_M$
  $u^{n+1} := u_{l_M}^{n+1}$ and $S^{n+1} := S_{l_M,k_M}^{n+1}$.
end
```

Note that the full equation (12) is almost symmetrized and almost well conditioned. Namely, the condition number is of order $\mathcal{O}(D\Delta t/(\Delta x)^2)$. Hence, a diagonal preconditioner works well in practice, in contrast to elliptic equations where the coefficient matrix is ill conditioned and extensive research has been carried out to develop an efficient preconditioner.

We now outline generalizations of the above algorithm in several directions. First, it is easy to see that the above algorithm applies to problems with several interfaces. Second, we note that the procedure applies to multidimensional problems, as long as the adjustment in Step K3 is kept local in space to avoid introducing any spurious nonzero saturation to the location where the saturation is zero. Third, Because the

Figure 2 The saturation "jumps up" across the interface from a higher permeability zone to a lower permeability zone.



coefficient matrix for the pressure equation has a much bigger condition number than that for the saturation equation, it is much more expensive to solve equations in (1) than to solve equation (2) at each time step. Physically the Darcy velocity is much smoother and varies less rapidly than the saturation. Thus, we can use larger time steps for pressure equations in (1) and smaller time steps for the saturation equation (2) (see [DEW83, Ewi84] for details).

5 Numerical Experiments

In this section we present a numerical example to show the promise of the algorithm. More extensive results can be found in [Ers96]. In the example, the space domain $(a, b) = (0, 1)$ with the interface located at $d = 0.5$. The time interval $[0, T] = [0, 0.048]$, $\varepsilon = 0.01$, $\lambda_w(S) = S^2$, $\lambda_o(S) = (1 - S)^2$, $\Delta x = 1/150$, $\Delta t = 0.001$, $K = 10$ on $(0, 0.5)$ and 1 on $(0.5, 1)$. The initial condition is an established shock given by

$$S_0(x) = \begin{cases} 1 - \dfrac{0.3}{0.4}x, & \text{if } 0 \leq x \leq 0.4, \\ 0, & \text{if } 0.4 < x \leq 1. \end{cases} \tag{13}$$

In the numerical experiments, $l_M = 1$ and $k_M = 4$. Namely, we extrapolated the current velocity field u^{n+1} by its values at the previous time steps and did not iterate on equations in (1). With the extrapolated velocity field at the current time step, we iterated four times on the saturation equation (2) at each time step. It was seen in Figure (1) that the permeability has considerable effect on diffusion and capillary pressure. For a fixed saturation the capillary pressure is higher in a lower permeable zone than it is in a high permeable zone. We observe that the continuity of capillary pressure in (4) enforces a jump up in the saturation profile across the interface. The numerical results are free of oscillation or numerical dispersion, and agree with the results in [CY92].

Acknowledgement

This research was supported in part by DE-FG05-95ER25266, ONR N00014-94-1-1163, the Royal Norwegian Council for Scientific and Industrial Research (NTNF), the Research Council of Norway (NFR), and the University of Bergen.

REFERENCES

[BW86] Bjørstad P. E. and Widlund O. B. (1986) Iterative methods for the solution of elliptic problems on regions partitioned into substructures. *SIAM J. Numer. Anal .*

[CJ86] Chavent G. and Jaffe J. (1986) *Mathematical models and finite elements for reservoir simulation.* North-Holland, Amsterdam.

[CY92] Chang J. and Yortsos Y. C. (1992) Effect of capillary heterogeneity on buckley-leverett displacement. *SPE Reservoir Engineering .*

[DEES90] Dahle H. K., Espedal M. S., Ewing R. E., and Sævareid O. (1990) Characteristic adaptive sub-domain methods for reservoir flow problems. *Numerical Methods for PDE's .*

[DEW83] Douglas J., Ewing R. E., and Wheeler M. F. (1983) The approximation of the pressure by a mixed method in the simulation of miscible displacement. *R.A.I.R.O. Analyse Numerique .*

[DR82] Douglas J. and Russell T. F. (1982) Numerical methods for convection-dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures. *SIAM J. Numer. Anal .*

[EE87] Espedal M. S. and Ewing R. E. (1987) Characteristic petrov-galerkin subdomain methods for two-phase immiscible flow. *Comput. Meth. Appl. Mech. Engrg .*

[Ers96] Ersland B. (1996) *On numerical methods for including the effect of capillary pressure forces on two-phase, immiscible flow in a layered porous medium.* PhD thesis, University of Bergen.

[Ewi84] Ewing R. E. (ed) (1984) *Research Frontiers in Applied Mathematics.* SIAM, Philadelphia.

[SBG96] Smith B. F., Bjørstad P. E., and Groop W. D. (1996) *Domain decomposition: parallel multilevel methods for elliptic partial differential equations.* Cambridge University Press.

[WDE$^+$94] Wang H., Dahle H. K., Ewing R. E., Lin T., and Våg J. E. (1994) Ellam-based domain decomposition and local refinement methods for advection-diffusion equations with interfaces. In *Contemporary Mathematics, Vol 180.* American Mathematical Society.

99

Cyclic Symmetry in Geometrical Nonlinear Analysis of Structures

Andrei Vasilescu

1 Introduction

A possible challenge in the work of a research or designer is when a structure with cyclic period geometry appears. This type of symmetry remains an up-to-date subject in the theoretical and practicable field of engineering. The cyclically symmetry is present in many civil engineering structures (domes, cooling towers, chimneys, etc.) or in mechanical engineering (milling cutters, turbine bladed disks, gears, fan or pump impellers, etc.).

Such a structure may be considered as a domain composed by identical, coupled subdomains positioned symmetrically with respect to an axis. Analysis of one of the subdomains, named fundamental, and its high degree of repetition represents the key in obtaining major savings in calculus.

The symmetry of a state of a structural system is an intrinsic property that is independent from the external loading or of the analysis type: linear or nonlinear, static or dynamic, etc. Specific methods have been developed and implemented in FEM programs (MSC NASTRAN, ANSYS, PAFEC, etc.) **only** in the linear elastic and modal analysis. The published literature on cyclic symmetry presented the advantages of these specific methods (e.g. an important reduction of the computational effort comparative with the common approach). However, the aim of this paper is to survey the main directions possible to use the initial cyclic symmetrical configuration in a geometrical nonlinear analysis of structures and to present an example on a Schwedler dome.

2 Cyclical Symmetry

The cyclical symmetry was used in a FEM application for the first time by Richard Courant in his historical paper [Cou43] to reduce the effort of his "numerical treatment of the plane torsion problem for multiple-connected domains" [Vas94]. The mathematician Hermann Weyl presented in his essay [Wey52] the symmetry in nature

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

(from the cell in biology to the mineral crystal) and art closely related to some aspects of the mathematical theory of groups and subgroups. This group theory, especially the representation theory, is penetrating now in the FEM domain.

It is relevant to present the overall stiffness matrix $\mathbf{K}$ for a cyclically symmetric structure. It is essential to the development of the theory that the reference system of coordinates should itself be cyclically symmetric: the axes attached to a subdomain are carried into those of the next similar subdomain when the whole domain (the structure) is rotated through an angle of $2\pi/N$. N is the number of identical subdomains and it represents the order of cyclic symmetry. In this way the overall stiffness matrix has a special form: it is quasicirculant or block circulant. The theory of circulant and quasicirculant matrices shows that $\mathbf{K}$ is similar to a matrix with diagonal blocks [Dav79]. This is also justified for the mass matrix $\mathbf{M}$, damping matrix $\mathbf{C}$, etc. of full structure.

The immediate consequence is that the initial problem, for any form of matrix analysis, is multiplied N times, but each of them is drastically reduced in size. This reduction is the main advantage to be considered in the following because the cost of solving a problem rises much faster than its size. For instance, the mathematical representation of the model in terms of physical quantities leads to the system of simultaneous linear equations in the static analysis

$$\mathbf{KU} = \mathbf{P} \tag{2.1}$$

where $\mathbf{U}$ is the displacement vector and $\mathbf{P}$ is the nodal force vector. This problem can be split into N decoupled problems. Hence, there are preferable to solve many small sets of linear algebraic equations, quickly and involving less computer memory, using common matrix manipulation capabilities. For the decoupled eigenvalue problems there is another advantage: the condition number of each matrix from reduced eigenvalue problem is less or equal to the condition number of the overall matrix from uncoupled problem, respectively. Other advantage is that the analyst will be required to input data for the fundamental domain (geometry, material properties, connections, boundary conditions, etc.). Transformation of mathematical representation of the model from physical components to cyclic components is possible taking into account that the deflections of a subdomain j are related to the adjacent subdomains $j+1$ by a complex constant [Tho79].

$$\mathbf{U}_j = \mathbf{U}_{j+1} e^{i\mu} \tag{2.2}$$

where $\mu = 2\pi n/N$ is propagation constant, $n = 0,1,2,\ldots\mathrm{Int}\left(\frac{N}{2}\right)$. The force acting in the k-th degree of freedom (d.o.f.) of the j-th subdomain may be expressed by discrete Fourier transform [BSR91] as:

$$p_{jk} = \sum_{n=1}^{N} f_{nk} e^{-i(j-1)\mu} \tag{2.3}$$

The cyclic component of force for the k-th d.o.f. is obtained by inverse transform:

$$f_{nk} = \frac{1}{N}\sum_{j=1}^{N} p_{jk} e^{i(j-1)\mu} \tag{2.4}$$

Since all forces acting on adjacent subdomains are related by the same phase constant, $e^{-i\mu}$, the deflection on adjacent subdomains should be connected in the same way [Tho79]:

$$u_{jk} = \sum_{n=1}^{N} d_{nk} e^{-i(j-1)\mu} \tag{2.5}$$

Using these equations in first equation, the decoupled matrix equation for various harmonics is given by

$$\mathbf{K_n D_n = F_n} \tag{2.6}$$

The application of complex constraints produces a fully populated complex stiffness matrix $\mathbf{K}_n$ and a set of simultaneous complex equation is to be solved for each harmonic, n.

After the solution phase of the problem, it is necessary to transform the results from cyclic components back into physical components. The obtained displacement vector $\mathbf{D}_n$ may be transform to cover displacements on the entire domain. The displacement of the k-th d.o.f. of the j-th subdomain is calculate [BSR91] for N odd:

$$u_{jk} = d_{0k} + 2\sum_{n=1}^{(N-1)/2} [Re(d_{nk})\cos\{(j-1)\mu\} + Im(d_{nk})\sin\{(j-1)\}\mu]\,; \tag{2.7}$$

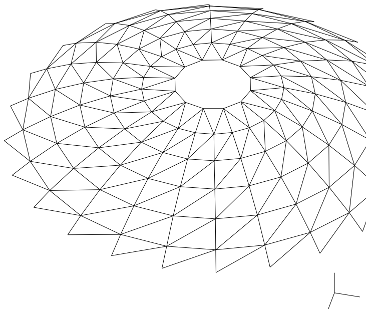
for N even:

$$u_{jk} = d_{0k} + (-1)^{(j+1)} d_{\frac{N}{2}k} + 2\sum_{n=1}^{\frac{N}{2}-1} [Re(d_{nk})\cos\{(j-1)\mu\} + Im(d_{nk})\sin\{(j-1)\}\mu] \tag{2.8}$$

Similar relations may be used for other physical components like stress, temperature, etc.

Any more savings in calculus are possible to obtain if the loading is also cyclical symmetric of the same order N. Then each subdomain is assumed to be loaded identically. It follows that the corresponding nodal displacements from the two boundaries of the fundamental domain are identically in the radial and circumferential direction, respectively [ZS72]. Since the constraints between two boundaries are of real type, will be necessary to solve only one system of simultaneous linear equations written for the fundamental domain [Vas95].

Figure 1



3 Geometric Nonlinear Analysis

In linear structural analysis, it is assumed that the joint displacements of the structure under the applied loads are negligible with respect to the original joint coordinates. Thus, the geometric changes in the structure can be ignored and the overall stiffness of the structure in the deformed shape can be assumed to equal the stiffness of the undeformed structure. However, in the space truss structures like domes, significant changes in the initial geometry can occur. In such a case, the stiffness of the dome in the deformed shape should be computed from the new geometry of the structure. That means to formulate the condition of equilibrium in the deformed configuration, since the truss members of the dome are assumed to have linear stress-strain relationships.

The problem to be solved in static geometric nonlinear structural system is the determination of the displacements, $\mathbf{U}$, corresponding to some load, $\mathbf{P}$, since the stiffness matrix, $\mathbf{K}$, in the equilibrium equation is a function of the joint displacements $\mathbf{U}$, which are as yet unknown. The analysis will typically proceed in two phases: the first is a solution which may be termed the incremental phase; the second is a corrective procedure applied to the first in an attempt to obtain the solution close to the equilibrium path.

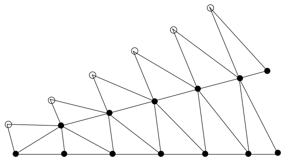
The way to preserve the advantage induced from the initial cyclic symmetry of the structure is to choose the modified Newton-Raphson method (MNR), in which the same matrix $\mathbf{K}_{(0)}$ is used for all the iterations. $\mathbf{K}_{(0)}$ is the stiffness matrix from the linear analysis, named also the tangent stiffness matrix. This matrix is computed at the beginning, and has a circulant or quasicirculant form.

4 Solution Algorithm

The advantages of the cyclic symmetry techniques are embedded in solving the linear equations' system. In the common geometrical nonlinear analysis, the global stiffness matrix $\mathbf{K}_{(0)}$ can be constructed only once, from the stiffness matrices of the individual members of the structure by the general assembly procedure.

In particular, the cyclic symmetry capabilities enable to solve the linear equations' system without assembling $\mathbf{K}$ - only the stiffness matrices of an appropiate fundamental domain are required. The algorithm for this method is summarized in the following:

Figure 2



Implementation

1. Initialization and input parameters such as geometric and material properties, connectivity, boundary conditions, and reference loads.

2. Initial step: solve $\mathbf{K}_{(0)}\mathbf{U}_{(0)} = \mathbf{P} \Rightarrow$ get $\mathbf{U}_{(0)}$.

3. Compute initial member end forces in global coordinates.

4. For each iteration (n)

4.1. For each element, form its stiffness matrix in local and then, in global coordinates $\mathbf{k}_{(n)}=\mathbf{k}(\mathbf{U}_{(n-1)})$.

4.2. Compute unbalanced joint loads in global coordinate system $\mathbf{P}_{(n)} = \mathbf{P}$ - $\mathbf{K}_{(n)}\mathbf{U}_{(n-1)}$.

4.3. Solve $\mathbf{K}_{(0)}\triangle\mathbf{U} = \mathbf{P}_{(n)} \Rightarrow$ get $\triangle\mathbf{U}$.

4.4. Update joint coordinates $\mathbf{U}_{(n)} = \mathbf{U}_{(n-1)} + \triangle\mathbf{U}$

4.5. Check convergence by considering force norm. If no convergence and $n <$ max. number of iterations GO TO Step 4.1.

5. Print on files final results.

6. Stop.

Practical Considerations

The type of the conservative structure considered in this approach is a cyclically symmetric space truss. In the iterative procedure, the stiffness matrices of the truss elements are updated corresponding to the new joint coordinates. The updating of initial length for each element is used to calculate the rotation matrix. This improves the convergence of the iterative process.

Apparently, we must assemble the overall tangent stiffness matrix of the structure to calculate in each iteration the unbalanced forces (see Step 4.2). This would compromise the present method as it is extremely expensive in the memory allocation and CPU time. In fact, we need to process only the new element stiffness in local and then in global coordinates. The unbalanced forces in global coordinate system are established by contribution of each element and is no need to assemble the overall tangent stiffness matrix of the structure.

Schwedler Dome Example

A computer program has been developed in FORTRAN 77 language to perform geometrical nonlinear analysis on cyclically symmetrical space truss structures. The structure analyzed was a spherical dome (25.6 m radius) of Schwedler type. The order of cyclic symmetry is 12. It can be observed that no plan of symmetry exists. Some different constructive solutions were studied to cover a cylindrical oil tank (32.0 m diameter). The constructive details (geometry, bar sections, loading, etc.) are from [SR85]. Since the whole structure (see Figure 1) is made up of 408 trusses connected in 166 nodes (396 active d.o.f.), the fundamental subdomain to be analyzed (see Figure 2) contains 34 trusses pinned in 20 nodes. In this case, the size of the problem is given by the 33 active d.o.f. on the subdomain after the boundary constraints were applied. We need only 6 iterations to obtain the specified convergence (0.01) of unbalanced force norm.

5 Conclusions

A solution strategy for the geometrical nonlinear analysis of elastic cyclically symmetric structures like dome have been presented. The formulation has been adopted the modified Newton-Raphson method (MNR) to take full advantage of cyclic symmetry of the structure. We remember that in nonlinear analysis solution algorithms are problem dependent. It is well known, however, that the convergence of the MNR method is linear and may run into serious difficulties if the loading level produces large changes in displacements. A simple method to prevent this is to scale back the load. At this stage, it is necessary to underline the limit of the MNR method in the cyclically symmetric context: it works on mild geometrical nonlinear structures like domes.

6 Acknowledgement

The author is very grateful to Petter Bjørstad and Emmanuel Vavalis for their useful suggestions. The presentation of this work has partly been supported with funds from the Organizing Committee of the DD9 Conference.

REFERENCES

[BSR91] Balasubramanian P., Suhas H., and Ramamurti V. (1991) Skyline solver for the static analysis of cyclic symmetric structures. *Computers & Structures* 38(3): 259–268.

[Cou43] Courant R. (1943) Variational methods for the solution of problems of equilibrium and vibrations. *Bull. Amer. Math. Soc.* 49: 1–23.

[Dav79] Davis P. (1979) *Circulant Matrices.* John Wiley & Sons.

[SR85] Soare M. and Raduica N. (1985) A comparation of the structural efficiency of some braced domes. *Space Structures* 1: 137–142.

[Tho79] Thomas D. (1979) Dynamics of rotationally periodic structures. *International Journal for Numerical Methods in Engineering* 14: 81–102.

[Vas94] Vasilescu A. (1994) The Symmetry and the Courant Element. In M. Krizek P. N. and Stenberg R. (eds) *Finite element methods: fifty years of the Courant method*, number 164 in Lecture Notes in Pure and Applied Mathematics, pages 461–465. Marcel Dekker, Inc., New York, Basel, Hong Kong.

[Vas95] Vasilescu A. (May 1995) On the Advantages in Periodical Structures Analysis. In Ieremia M. and Berbente C. (eds) *Proc. Third Int. Conf. on Boundary and Finite Element, ELFIN3*, pages 238–247. SIAC, Constanta.

[Wey52] Weyl H. (1952) *Symmetry.* Princeton University Press.

[ZS72] Zienkiewicz O. and Scott F. (1972) On the principle of repetability and its application in analysis of turbine and pump impellars. *International Journal for Numerical Methods in Engineering* 9: 445–448.

100

Dynamic Scheduling of Substructure Computations in an Industrial Production Environment

Petter E. Bjørstad, Jon Bækhus, and Jeremy Cook

1 Introduction

Domain decomposition of large structural analysis problems based on physical substructures modeled separately can be used to get a good ordering for the direct Cholesky factorization of the global stiffness matrix [Prz85], [SBG96]. For large substructure elimination trees one can further increase the computational speed by processing independent substructures in parallel [BBH91]. However, this technique often results in substructures of very different size leading to an important load balancing problem. This paper discusses experience in an industrial setting using a workstation cluster.

The SESAM[1] software package is a general, structural analysis finite element package for static and dynamic linear analysis. The linear equations are solved by a module SESTRA using a multilevel superelement technique [Det94]. SESAM has users worldwide with focus on the analysis of ships and offshore structures. A failure in such structures can have very serious consequences. Thus, the demand for more accurate analysis is a driving force towards efficient and more robust computer implementations.

The stiffness matrix of a superelement (which here always corresponds to a substructure of the full structure) takes the form

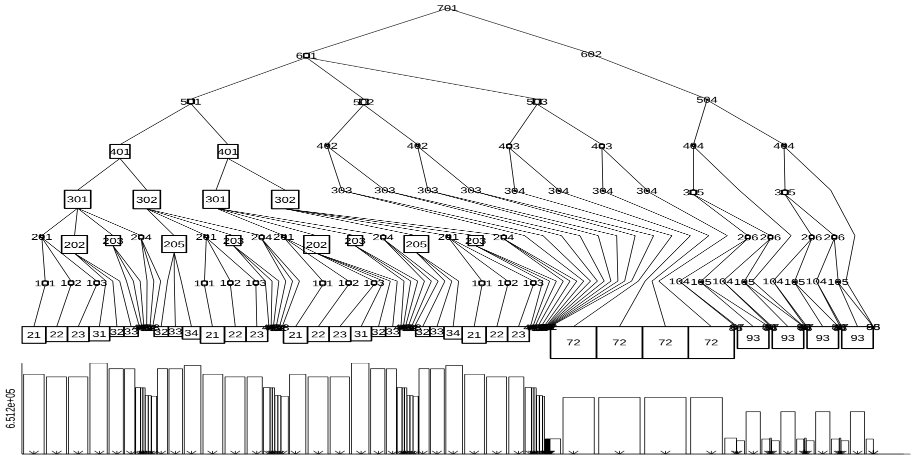
$$\begin{pmatrix} K_{ii} & K_{ir} \\ K_{ir}^T & K_{rr} \end{pmatrix}. \tag{1.1}$$

Here i denotes an internal node while r is a node on the substructure boundary (connecting it to other substructures). The computation lends itself naturally to a two-level parallel implementation, a coarse grain level where independent substructures can be computed in parallel and a fine grain level where the linear algebra within a substructure calculation is parallelized. This calculation consists of forming the Schur

[1] SESAM is developed and marketed by Det Norske Veritas a/s.

Ninth International Conference on Domain Decomposition Methods
Editor Petter E. Bjørstad, Magne S. Espedal and David E. Keyes

Figure 1 The complete substructure tree of the Troll model. The tree has 147 tasks and must be used in the dynamic back substitution algorithm. There are many identical tasks and only 48 are needed in the scheduling of the matrix factorization. The critical paths are shown in the bar graph below the tree. The size of the boxes around each superelement indicates the computational load.



complement

$$S = K_{rr} - K_{ir}^T K_{ii}^{-1} K_{ir}. \tag{1.2}$$

In practice, this is carried out by a Cholesky factorization of the interior part K_{ii}, a forward triangular solve with K_{ir} as the right hand side, and a matrix multiplication where the resulting matrix K_{rr} is symmetric. We overwrite K_{ir} in this process and note that we only compute the Schur complement for substructures that are different. The same substructure can be used as a building block in several places when forming the global model with a considerable computational savings, see Figure 1.

The overall parallel strategy was outlined in [Bjø87] and the coarse grain part was implemented and turned into a commercial software product by Hvidsten [Hvi90]. A fine grain parallel implementation is described in [CBB96].

In this work we are only interested in testing the coarse grain parallel implementation in an industrial setting in order to report on the actual benefit of using a cluster of workstations. We will further report on the most important factors that must be taken into account in order to get satisfactory parallel performance. We have chosen a finite element model of the Troll[2] offshore gas platform as our test case. The model of Troll that we will use is of medium size having 701508 degrees of freedom representing the 369 meter tall concrete base of the platform.

[2] Troll went into production in 1995, with a height of 472 meters and a total weight of more than 10^9 kg it is the largest structure made by man and subsequently moved to its final destination. See http://www.shell.no/troll/ for pictures and more information.

2 Scheduling of Parallel Tasks

We use a dynamic scheduling algorithm as opposed to an a priori statically determined scheduling. Each time a processor is available and requests a new task, SESTRA will execute its scheduling algorithm. Two factors must be carefully examined by this algorithm:

- The computational size of a task
- The computational power of a workstation.

The size of a task is available as an estimate of the required number of floating point operations to form the Schur complement given in (1.2). This estimate is derived from the number of internal and boundary degrees of freedom where also the bandwidth of K_{ii} and empirical factors to compensate for fill during the factorization are included. Ideally, this estimate could be even more precise if based on a symbolic factorization phase, but this would require accessing the data twice which may carry a significant cost. Also, the amount of disk space needed to store the superelement matrices in factored form must be estimated. Thus, the computational task is estimated with respect to both time and space.

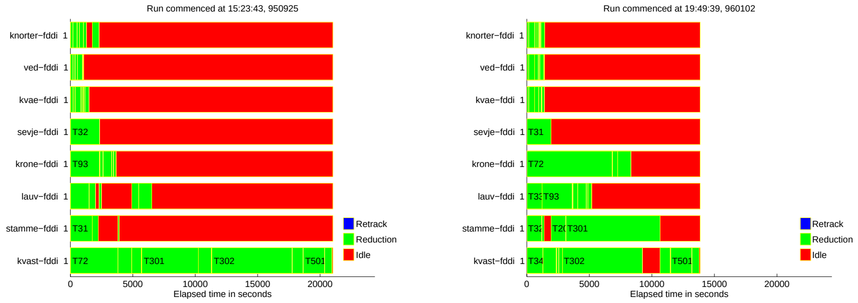
The power of a workstation closely matches the task estimate. The algorithm is given a list of 'realistic' floating point speeds of each workstation. Since this speed unlike the 'peak speed' often quoted by vendors, depends on actual workstation load it may even be updated dynamically by comparing actual elapsed time with the predicted one after each task completion. The space variable of a workstation corresponds to the available local disk space and the size of its memory. If the substructure cannot be stored locally, then this can greatly affect I/O speed and thereby overall elapsed time for a given calculation. Similarly, a large memory will be used to cache disk I/O on all modern workstations. Thus, this factor can have a significant impact on the computing time as well.

In our first algorithm, the largest available task was scheduled first. It turned out that this heuristic was too primitive and our current algorithm uses the estimated critical path with respect to computational loads as the basis for task scheduling. That is, the possible tasks will always be considered starting with the substructure which is at the end of the longest critical path to the root of the tree. The algorithm will also keep a table holding the estimated time until each workstation has finished its current task. Informally stated, the scheduling proceeds as follows:

MY workstation is idle and requests a new task:

1. Select the task at the end of the current critical path
2. Estimate the completion time for this task on MY workstation
3. Estimate the completion time for all other (busy) workstations on this task, taking into account the estimate for completion of their current work
4. If I complete first, assign this task to ME and exit here, otherwise update the completion estimate of whoever will be faster than ME with this task
5. If there are more available tasks go to 1
6. If no task found for ME, go idle and check back later.

Figure 2 Scheduling of superelement tasks on 8 nodes. To the left, the time history from September 95, with the largest task first and too much priority to the fastest processor. To the right, the time history from January 96 with priority to tasks on the critical path.



3 Experience with the Scheduling

In this section we discuss our test problem using the dynamic scheduling of substructures as outlined above.

Our cluster of workstations consists of eight DEC Alpha EV45 with 233 MHz processors. One machine, kvast, has 512 MB memory while the other seven have 128 MB. Each machine has a separate FDDI segment all of which are interconnected by a Gigaswitch. Four machines each have 4 GB of local disk while the other four have only one gigabyte of local disk that can be used in the computation. On the single 512 MB workstation the analysis of Troll takes about 9 hours while the time increases to 14 hours on a 128 MB machine with 4 GB of local disk. Thus, SESTRA runs almost 60 percent faster on the machine with large memory due to efficient caching of file I/O to the free memory. The cluster is therefore inhomogeneous with respect to our scheduling algorithm despite the fact that each processor has the same theoretical speed. A substructure task will always use a disk that is large enough to hold its local files. Our algorithm from the previous section will rate the 512 MB machine as more powerful and dynamically further adjust the power of the machines depending on whether they need to access a local or remote disk for their current task.

The substructure tasks from Troll are of different sizes. The smallest task takes 9 seconds to complete, whereas the largest needs about two hours on the fastest machine. That is, we have about a factor 1000 in difference between the largest and the smallest tasks in this model.

The improvement from always attacking the largest available task to using a set of updated critical paths as a basis for scheduling can be seen in Figure 2. The left picture shows a situation where the largest task is scheduled first and the most powerful workstation ends up with too much work. We observe that all the tasks on the path after 72 are quite light, while tasks 301 and 302 higher up in the tree are quite heavy. The slower machines do not participate as much as they could and the total analysis time is 5 hours and 50 minutes. The right part of the figure shows a much better distribution resulting from our algorithm given earlier. The total time has been reduced by 34 percent to 3 hours and 51 minutes.

Table 1 Elapsed time for the factorization and back substitution phase.

Workstations	Matrix Factorization	Back Substitution
1	8h 57m	2h 55m
2	6h 05m	-
4	4h 03m	1h 32m
8	3h 51m	0h 54m

Table 1 shows the elapsed time with our scheduling algorithm for different number of participating workstations. The gain from using more than 4 workstations is only slight, and is clearly not cost effective. There are only a limited number of branches with a significant amount of computation so adding more processors does not help unless we also exploit the finer grain parallel work when computing (1.2) inside each substructure task. However, for many customers having 2 to 8 workstations available, this improvement means that they can carry out two analysis each day instead of one overnight run.

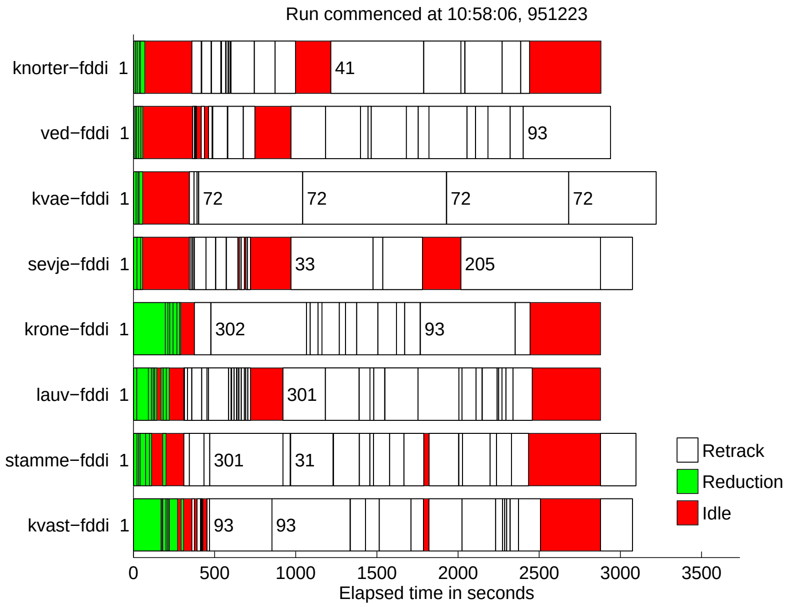
Often current jobs are larger than Troll, requiring more than 25 hours on a single workstation. This may be reduced to less than 10 with a 4 workstation cluster.

4 Parallel Back Substitution and New Loads

The main purpose of structural analysis calculations is to study how a given structure reacts to different loading conditions such as wind, wave, surface or simply point loads. Often, the main interest is focused on the response of specific substructures only. After the factorization phase has been completed such questions can be studied with considerable flexibility. Given a new right hand side (load condition), the analysis will first again proceed up the tree in Figure 1 starting at the substructure(s) subject to the load in order to transform the right hand side consistently with the elimination of interior degrees of freedom when we transformed the original system from equation (1.1) to (1.2) during the factorization step. This calculation involves a triangular solve and a matrix vector multiplication within each task along the path. In engineering terms this is called load reduction. Next, we must transverse the tree back, starting from the root and again do a back substitution and a matrix vector multiplication for each substructure along the path to the substructure whose response we are interested in. This process differs from the factorization step by:

- Repeated occurrences of substructures are now separate tasks. The number of tasks in our test model increases from 48 to 147.
- When traversing the tree from the root to the leaves, we get more and more parallel tasks as we go.

Figure 3 Task scheduling for retracking



- The size of each task depends more weakly on the number of equations. It is therefore easier to achieve good load balancing.
- All tasks grow linearly with the number of load cases.
- Repeated superelements cannot access the same substructure data simultaneously. Dynamic scheduling should therefore give priority to repeated tasks.
- The algorithm used in Figure 3 just searches a list of available tasks and grabs the first available, but more advanced logic have also been implemented.

We report on a case with complete back substitution for the entire structure with 20 different load conditions. The computing time is given in Table 1 and shows good utilization of all eight workstations. We note that this part of the computation is about 30 percent of the factorization phase, thus this time can be even more significant in an analysis that may have several hundred different loading conditions. From Figure 3 we see that even a very simple dynamic scheduling algorithm works well in this case.

5 Conclusions

We have documented the benefits of dynamic substructure scheduling within a large commercial structural analysis code. Our scheduling achieves near optimal results for the example tested in the factorization phase. This approach automatically incorporates knowledge about the actual load present in the workstation environment. The back substitution phase has excellent parallel properties when there are many right hand sides and a substantial part of the complete solution is wanted.

We have shown that a large industrial application, traditionally run on large supercomputers, can be used successfully for *industrial strength* problems in a workstation environment giving a very cost effective solution. Using a workstation environment also carries the added benefit that large result files can be postprocessed on the same architecture.

Acknowledgement

The authors express thank to A/S Norske Shell, the development operator of Troll GAS for permission to use the Troll GBS finite element model. The model was made by Norwegian Contractors before building the platform's gravity base structure. This work was carried out with funding from the EC-ESPRIT III EUROPORT subproject 8586.

REFERENCES

[BBH91] Bjørstad P. E., Brækhus J., and Hvidsten A. (1991) Parallel substructuring algorithms in structural analysis, direct and iterative methods. In Glowinski R., Kuznetsov Y. A., Meurant G. A., Périaux J., and Widlund O. B. (eds) *Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 321–340. SIAM, Philadelphia, PA. Proceedings from the fourth international symposium on domain decomposition, Moscow May 1990.

[Bjø87] Bjørstad P. E. (1987) A large scale, sparse, secondary storage, direct linear equation solver for structural analysis and its implementation on vector and parallel architectures. *J. Parallel Comp.* (5).

[CBB96] Cook J., Bjørstad P. E., and Brækhus J. (April 1996) Multilevel parallel solution of large, sparse finite element equations from structural analysis. In Liddel H., Colbrook A., Hertzberger B., and Sloot P. (eds) *High-Performance Computing and Networking*, volume 1067, pages 404—412. Springer. Lecture Notes in Computer Science.

[Det94] Det Norske Veritas – Sesam, P.O. box 300, N-1322 Høvik, NORWAY (March 1994) *SESAM technical description.*

[Hvi90] Hvidsten A. (1990) *On the Parallelization of a Finite Element Structural Analysis Program.* PhD thesis, Computer Science Department, University of Bergen, Norway.

[Prz85] Przemieniecki J. S. (1985) *Theory of Matrix Structural Analysis.* Dover Publications, Inc., New York. Reprint of McGraw Hill, 1968.

[SBG96] Smith B. F., Bjørstad P. E., and Gropp W. (1996) *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press.