R. Hiptmair
L. Scarabosio
E. Fonn

Spring Term 2013

Numerical Methods for Partial Differential Equations

ETH Zürich
D-MATH

Homework Problem Sheet 9

Introduction.

Problem 9.1 Trace Error Estimates

This problem serves to brush up your familiarity with a priori finite element convergence estimates. It should also give you an idea what these estimates can be used for, for instance for the prediction of the rate of algebraic convergence of derived quantities.

On a convex polygon $\Omega \subset \mathbb{R}^2$ with exterior unit normal $\boldsymbol{n}$ we consider the boundary value problem

$$\begin{aligned} -\Delta u &= f \quad \text{in } \Omega, \\ \boldsymbol{n} \cdot \operatorname{grad} u + u &= 0 \quad \text{on } \partial\Omega, \end{aligned} \tag{9.1.1}$$

with source function $f \in L^2(\Omega)$.

(9.1a) State the variational formulation of (9.1.1).

HINT: Do not forget the appropriate function spaces.

(9.1b) Discuss existence and uniqueness of solutions of the variational formulation obtained in subproblem (9.1a).

(9.1c) The boundary value problem (9.1.1) is discretized by means of linear Lagrangian finite elements on a sequence $(\mathcal{M}_i)$ of triangular meshes of Ω created by uniform regular refinement of a coarse mesh $\mathcal{M}_0$. Let $u_i \in \mathcal{S}_1^0(\mathcal{M}_i)$ denote the finite element solution on mesh $\mathcal{M}_i$.

Quantitatively, predict the asymptotic behavior of the error norms $|u - u_i|_{H^1(\Omega)}$ and $\|u - u_i\|_{L^2(\Omega)}$ in terms of $N_i := \dim \mathcal{S}_1^0(\mathcal{M}_i)$.

(9.1d) In the setting of the previous sub-problem, what asymptotic behavior can be expected from the error norm $\|u - u_i\|_{L^2(\partial\Omega)}$ on the boundary in terms of $N_i := \dim \mathcal{S}_1^0(\mathcal{M}_i)$.

HINT: Use the multiplicative trace inequality [NPDE, Thm. 2.9.7].

(9.1e) The LehrFEM function `solveImpedanceBVP(mesh)` (provided in the file `solveImpedanceBVP`
solves (9.1.1) for $f \equiv \cos(\|\boldsymbol{x}\|)$ on a polygon using linear Lagrangian finite elements on triangular meshes. The domain is specified through the argument `mesh` that passes a basic LehrFEM mesh data structure as returned from `load_Mesh`.

Extend this code such that it returns the quantity $B(u_N) := \int_{\partial\Omega} u_N \, \mathrm{d}S$, where u_N is the finite element solution.

(9.1f) Measure the rate of convergence of $B(u_N) \to B(u)$ by computing the numerical values on meshes obtained by successive refinement of the triangular mesh (of a regular hexagon) stored in `Coord_Hexagon.dat` and `Elem_Hexagon.dat`. Do this in a MATLAB script `convergence.m`.

HINT: The "exact" value is $B(u) = 2.08154105279$.

HINT: To refine a mesh, use the LehrFEM function `refine_REG`.

(9.1g) Prove that

$$B(u) = \int_\Omega f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}. \tag{9.1.2}$$

HINT: Use the variational formulation.

Listing 9.1: Testcalls for Problem 9.1

```
mesh = load_Mesh('Coord_Hexagon.dat', 'Elem_Hexagon.dat');
mesh = add_Edges(mesh);
mesh.BdFlags = [zeros(6,1); -ones(6,1)];
mesh.ElemFlag = zeros(6,1);
solveImpedanceBVP(mesh)
```

Listing 9.2: Output for Testcalls for Problem 9.1

```
>> test_call

B =

    2.0815
```

Problem 9.2 Loss of Regularity Due to Mixed Boundary Conditions (Core problem)

[NPDE, Sect. 5.4] briefly reviewed elliptic regularity theory and identified reasons why the solution of a second order scalar linear elliptic boundary value problem may fail to belong to $H^2(\Omega)$ though the right-hand side function is in $L^2(\Omega)$. It was mentioned that one reason can be mixed boundary conditions, see [NPDE, Rem. 2.6.6]. This problem will supply the details and in doing so closely follows the consideration on "corner singular functions" presented in the course, see [NPDE, Eq. (5.4.4)].

As discussed in [NPDE, Sect. 5.4] the inevitable presence of corner singular function components in the solution of elliptic boundary value problems degrades the smoothness of their solutions (measured in terms of finite Sobolev norms, *cf.* [NPDE, Sect. 5.3.3]). This has direct impact on the rates of algebraic convergence of finite element solutions, and this will be explored in the second part of this problem by means of numerical experiments based on LehrFEM.

We consider a wedge-shaped domain

$$\Omega(\omega) = \{(r,\varphi) \mid r < 1,\ 0 < \varphi < \omega\}\,,$$

in polar coordinates, with $\partial\Omega = \Gamma_N \cup \Gamma_D$, where

$$\Gamma_N = \{(r,\varphi) \in \partial\Omega \mid \varphi = \omega\}$$

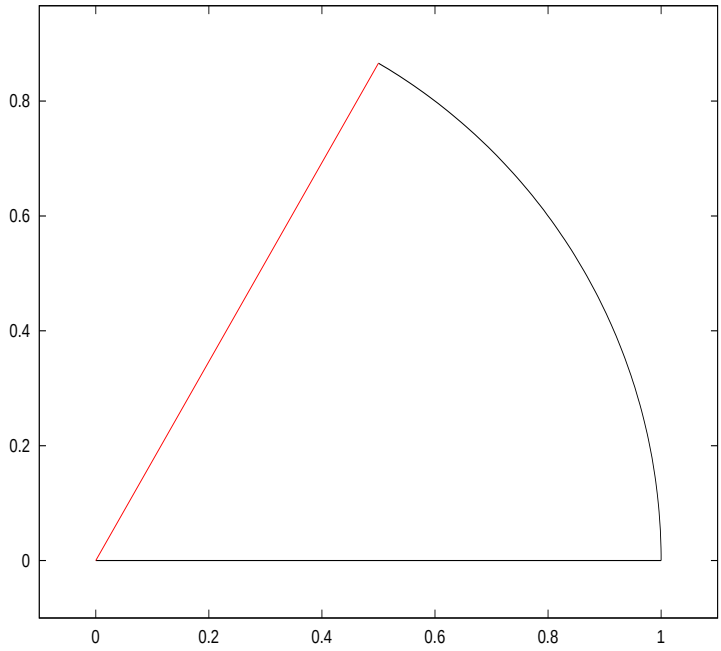

Figure 9.1: A sample domain for $\omega = \frac{\pi}{3}$ used in Problem 9.2

is the "left" angle of the wedge, and Γ_D contains the "right" angle and the curved part of the boundary. The domain is shown in Figure 9.2 for $\omega = \frac{\pi}{3}$. We also define

$$u_S(r,\varphi) = r^{\frac{\pi}{2\omega}} \sin\left(\frac{\pi}{2\omega}\varphi\right). \tag{9.2.1}$$

HINT: The use of suitable polar coordinates is recommended for the following subproblems, see [NPDE, Fig. 172], [NPDE, Eq. (5.4.5)], and [NPDE, Eq. (2.3.17)].

(9.2a) Show that $\Delta u_S = 0$ away from $r = 0$.

(9.2b) Show that $u_S \in L^2(\Omega)$ for $0 < \omega < 2\pi$.

(9.2c) Show that $u_S = 0$ on the x-axis, and $\operatorname{grad} u_S \cdot \boldsymbol{n} = 0$ on Γ_N.

(9.2d) Show that $u_S \in H^2(\Omega) \iff 0 < \omega < \frac{\pi}{2}$.

HINT: Remember that a formal way to verify that a function belongs to a certain Sobolev space is to confirm that its corresponding Sobolev norm is finite, that is, the defining integral makes sense (maybe only as an improper integral). Conversely, divergence of that integral indicates that the function is not contained in the Sobolev space.

(9.2e) Implement a MATLAB function

```
Mesh = bdflags(Mesh)
```

that takes a LehrFEM Mesh structure for the domain $\Omega = \Omega(\pi)$ (that is, the upper semi-disk), with edge and element information, and adds the boundary flags -1 for Γ_N and -2 for Γ_D.

HINT: An edge is part of Γ_N if both vertices are on the x-axis, with x-values between -1 and 0 inclusive. Otherwise, it is part of Γ_D. Note that `init_mesh` will not reproduce this geometry exactly, allow for a gracious epsilon (e.g. 10^{-8}) when doing the test, just take any value well above the mashine epsilon.

(9.2f) A MATLAB template is provided for a function

```
[N, l2, h1] = main(h)
```

that solves the mixed boundary value problem

$$\begin{aligned} -\Delta u &= 0 \quad \text{in } \Omega(\pi), \\ u &= u_S \quad \text{on } \Gamma_D, \quad \mathbf{grad}\, u \cdot \boldsymbol{n} = 0 \quad \text{on } \Gamma_N, \end{aligned} \tag{9.2.2}$$

by means of a Galerkin finite element method based on piecewise linear Lagrangian finite elements on a triangular mesh. This mesh (with approximate cell size h) should be generated by using the LehrFEM methods documented in [LehrFEM, Sect.1.2].The function should return (in `l2` and `h1`) approximate L^2-norm and H^1-seminorms of the finite element discretization error, as well as the number N of degrees of freedoms.

HINT: Of course, the exact solution is u_S. Use the MATLAB functions `cart2pol` and `pol2cart` for conversion between Cartesian and polar coordinates.

When generating the mesh, the call is

```
Mesh = init_Mesh(bbox, h, dist, @h_uniform, fixpts, display);
```

where

- `bbox` is a 2×2 matrix specifying two diagonally opposite corners of a *bounding box* that contains Ω, one point in each row.
- `h` is the initial approximate mesh size.
- `dist` is a signed distance function, see [LehrFEM, Sect.1.2].
- `@h_uniform` is the uniform cell size function, specifying that we want approximately the same cell sizes for the whole mesh.
- `fixpts` is an $N \times 2$ matrix specifying fixed points, i.e. points that must become vertices in the mesh. You will need to use this argument to fix the two corners and the origin, otherwise you might not be able to properly separate the two parts of the boundary.
- `disp` is a boolean flag. Set it to 1 if you want to watch the mesh generation process "live". This is useful for debugging. Otherwise, set it to 0.

An edge is part of Γ_N if both vertices are on the x-axis, with an x-value between -1 and 0 inclusive. Othwerwise, it is part of Γ_D.

In case of problems, a sample mesh stored in `coord_test.dat` and `elem_test.dat` has been provided. When run with this mesh (instead of the generated mesh), the function should output `N=76`, `l2=0.0192` and `h1=0.1877`.

HINT: To take into account the Dirichlet boundaries, use the Dirichlet boundary flag in `assemDir_LFE` as described in [LehrFEM, Sect.6.1]

(9.2g) Study the convergence of $|u_S - u_N|_{H^1}$ and $\|u_S - u_N\|_{L^2}$ in terms of the meshwidth. What rates of algebraic convergence do you observe?

HINT: Recall [NPDE, Rem. 1.6.22] and the use of the MATLAB `polyfit` function to perform linear regression.

Listing 9.3: Testcalls for Problem 9.2

```
fprintf('\n##bdflags');
clear Mesh
t = pi*(0:4)/4;
[x,y] = pol2cart(t,1);
Mesh.Coordinates = [0, 0; x', y'];
Mesh.Elements = [1 2 3; 1 3 4; 1 4 5; 1 5 6];
Mesh.ElemFlag = ones(size(Mesh.Elements,1),1);
Mesh = add_Edges(Mesh);
Mesh = bdflags(Mesh);
Mesh.BdFlags
```

Listing 9.4: Output for Testcalls for Problem 9.2

```
>> test_p2

##bdflags
ans =

    -2
     0
     0
     0
    -1
    -2
    -2
    -2
    -2
```

Problem 9.3 Debugging Finite Element Codes (Core problem)

[NPDE, Ch. 5] confronted you with theoretical results on the asymptotic convergence of finite element Galerkin solutions for 2nd-order elliptic boundary value problems. On the one hand, these estimates can be used to gauge the relative efficiency of different finite element approximations, as was discussed in [NPDE, Sect. 5.3.5]. On the other hand, expected rates of convergence are a fine probe for detecting errors in a finite element code. For instance, the observed convergence to a known analytic solution should match the theoretical predictions, unless the code is flawed. Another way of using the approximation results of [NPDE, Sect. 5.3.5] to identify a faulty finite element implementation is demonstrated in this problem.

Three different LehrFEM routines

$$\texttt{[A,phi] = assembleQFEX(mesh,f\_hd)}, \quad \texttt{X} \in \{1,2,3\}$$

purport to provide the Galerkin matrix and right-hand side vector for the finite element discretization of the variational problem

$$u \in H^1(\Omega): \quad \mathsf{a}(u,v) := \int_\Omega \operatorname{grad} u \cdot \operatorname{grad} v \,\mathrm{d}\boldsymbol{x} = \ell(v) := \int_\Omega f(\boldsymbol{x})v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \quad \forall v \in H^1(\Omega) \tag{9.3.1}$$

using *quadratic* Lagrangian finite elements (space $\mathcal{S}_2^0(\mathcal{M})$) on a triagular mesh $\mathcal{M}$ of some polygon $\Omega \subset \mathbb{R}^2$. The argument `mesh` is is supposed to pass a LehrFEM mesh data structure complete with edge information and element flags, whereas `f_hd` contains a handle to the source function f of type `@(x,varargin)`.

The routines return the Galerkin matrix and right-hand side vector for (9.3.1) w.r.t $\mathcal{S}_2^0(\mathcal{M})$ based on standard global shape functions of $\mathcal{S}_2^0(\mathcal{M})$, which are associated with interpolation nodes in the vertices and midpoints of edges. The following ordering of global shape functions is used: first we number the basis functions belonging to vertices based on the vertex array in the `mesh` data structure. Second, the basis functions associated with edges are ordered according to the numbering of the edges in the `mesh.Edges` field (see [NPDE, Ex. 3.5.18]).

(9.3a) Write a MATLAB function

```
Iu = interpolateQFE(mesh,u)
```

that accepts a LehrFEM mesh data structure `mesh` and a handle `u` to a real valued function u and returns the basis coeffcients of the nodal interpolant $\mathsf{I}_2 u \in \mathcal{S}_2^0(\mathcal{M})$. This particular piecewise quadratic interpolation is presented in [NPDE, Ex. 3.4.2].

(9.3b) Determine a sharp bound $T(h_\mathcal{M})$ in the estimate

$$|\mathsf{a}(u,u) - \mathsf{a}(\mathsf{I}_2 u, \mathsf{I}_2 u)| \le C T(h_\mathcal{M}) \,, \tag{9.3.2}$$

where $u : \overline{\Omega} \mapsto \mathbb{R}$ is supposed to be smooth and the unknown constant $C > 0$ may depend only on Ω and the shape regularity measure of $\mathcal{M}$.

Use the following result, which is a generalization of [NPDE, Cor. 5.3.32] to quadratic Lagrangian finite element spaces.

Theorem. *Let $\Omega \subset \mathbb{R}^d$, $d = 1,2,3$, be a bounded polygonal/polyhedral domain equipped with a simplicial mesh $\mathcal{M}$. Then the following interpolation error estimate holds for the nodal interpolation operator I_2 onto $\mathcal{S}_2^0(\mathcal{M})$*

$$\|u - \mathsf{I}_2 u\|_{H^1(\Omega)} \le C h^{\min\{3,k\}-1} |u|_{H^k(\Omega)} \quad \forall u \in H^k(\Omega), \quad k = 2,3 \,,$$

with a constant $C > 0$ depending only on k and the shape regularity measure $\rho_\mathcal{M}$.

HINT: Recall from [NPDE, Sect. 5.1] the arguments that suggested that the energy norm provides a relevant norm for measuring the discretization error.

(9.3c) Write a MATLAB function

```
enu = test_assembleQFE(mesh,assfn)
```

that computes $\mathsf{a}(\mathsf{I}_2 u, \mathsf{I}_2 u)$ for $u(\boldsymbol{x}) = \exp(\|\boldsymbol{x}\|^2)$ and the domain triangulated by the mesh described by the LehrFEM mesh data structure `mesh`. The argument `assfn` passes a handle to an assembly routine for quadratic Lagrangian finite element with the calling syntax of `assembleQFEX` introduced above.

HINT: Use the function `interpolateQFE` developed in (9.3b).

(9.3d) The file `squaremesh.mat` contains the LehrFEM mesh data structures for five increasingly refined triangular meshes of $\Omega = (0,1)^2$ in the variables `mesh1`, ..., `mesh5`. For each of the assembly routines `assembleQFEX(mesh,f)`, $\mathtt{X} \in \{1,2,3\}$ plot $|\mathsf{a}(u,u) - \mathsf{a}(\mathsf{I}_2 u, \mathsf{I}_2 u)|$ for these meshes and the function $u(\boldsymbol{x}) = \exp(\|\boldsymbol{x}\|^2)$ from (9.3c) against the meshwidth $h_{\mathcal{M}}$ in a suitable scale.

HINT: Use the function `test_assembleQFE` implemented in (9.3c). The mesh width $h_{\mathcal{M}}$ of a mesh stored in the LehrFEM data structure `mesh` can be computed by calling `get_MeshWidth(mesh)`.

Also, you may use

$$|u|^2_{H^1(\Omega)} = 23.7608$$

for $\Omega = (0,1)^2$.

(9.3e) Which implementations of the assembly routine are wrong, which are correct? Explain your answer.

Listing 9.5: Testcalls for Problem 9.3

```
fprintf('\n##interpolateQFE');
clear Mesh
Mesh.Coordinates = [0 0; 2 0; 1 1];
Mesh.Elements = [1 2 3];
Mesh.ElemFlag = ones(size(Mesh.Elements,1),1);
Mesh = add_Edges(Mesh);
Loc = get_BdEdges(Mesh);
Mesh.BdFlags = zeros(size(Mesh.Edges,1),1);
Mesh.BdFlags(Loc) = -1;
Mesh = add_Edge2Elem(Mesh);
interpolateQFE(Mesh, @(x)(sum(x.^2,2)))

fprintf('\n##test_assembleQFE');
test_assembleQFE(Mesh, @assembleQFE1)
```

Listing 9.6: Output for Testcalls for Problem 9.3

```
>> test_p2

##interpolateQFE
ans =

          0
    4.0000
    2.0000
    1.0000
    0.5000
```

```
    2.5000

##test_assembleQFE
ans =

  979.6587
```

Problem 9.4 Graded Mesh for the Poisson Equation

Example [NPDE, Ex. 5.2.9] sent the confounding message that lack of Sobolev regularity of the exact solution u of an elliptic boundary value problem thwarts higher rates of algebraic convergence for Lagrangian finite element solutions with higher polynomial degree p. This was later corroborated by the best approximation result of [NPDE, Thm. 5.3.42], which features the rate of convergence $\min\{p+1,k\}-1$ for the best approximation error in $\mathcal{S}_p^0(\mathcal{M})$ in the case of *h-refinement* and $u \in H^k(\Omega)$, see also [NPDE, Eq. (5.3.56)].

One wonders whether there is a way to recover fast convergence even if u lacks smoothness. The key is to abandon uniform h-refinement and switch to *adaptive local mesh refinement*. "Adaptive" indicates that this local mesh refinement has to take into account features of the solution. Thus fast convergence in terms of the problem size can be restored.

In this problem we will study a particular kind of local mesh refinement, known as *graded meshes* for a 2-point boundary value problem on $]0,1[$. In concrete terms, we consider the one-dimensional Poisson equation

$$-u'' = f \quad \text{in } \Omega =]0,1[\,, \quad u(0) = 0\,, \quad u(1) = 1\,, \tag{9.4.1}$$

where f is chosen such that the exact solution is $u(x) = x^\alpha$. We define a graded mesh $\mathcal{M} := \{]x_j, x_{J+1}[:\ j = 0,\ldots,M-1\}$ with grading factor β by the nodal points

$$x_0 = 0 < x_1 = \left(\frac{1}{M}\right)^\beta < \cdots < x_{M-1} = \left(\frac{M-1}{M}\right)^\beta < x_M = 1\,. \tag{9.4.2}$$

On this mesh, we perform the Galerkin finite element discretization of (9.4.1) by means of $\mathcal{S}_{1,0}^0(\mathcal{M})$, see [NPDE, Sect. 1.5.1.2].

(9.4a) Compute f as an analytic expression.

(9.4b) For which α is the solution $u \in H^1(\Omega)$ or $u \in H^2(\Omega)$?

HINT: This amounts to checking whether the corresponding Sobolev norms are finite.

(9.4c) Write the variational formulation for (9.4.1).

(9.4d) Compute the element matrix for (9.4.1) and piecewise linear Lagrangian finite elements for a generic cell $]x_{j-1}, x_j[$. Of course, we use the standard nodal basis consisting of "tent functions", see [NPDE, Eq. (1.5.52)].

(9.4e) Implement a MATLAB function

```
function Aloc = STIMA_Lapl_1D(Vertices)
```

that accepts a 2-vector of coordinates of the endpoints of a cell as argument and returns the element matrix for that cell.

(9.4f) Describe the sparsity pattern of the Galerkin matrix $\mathbf{A} \in \mathbb{R}^{M-1,M-1}$

(9.4g) In the file `main1D.m` you find a lacunary LehrFEM style MATLAB function for the solution, for fixed values of α and β, of the 2-point boundary value problem (9.4.1) using linear Lagrangian finite element on a graded mesh according to (9.4.2). The problem (9.4.1) is solved on several meshes with $M = 10 \cdot 2^l$, $l = 2, \ldots, 8$ (M is the number of intervals) and a convergence study for the H^1-seminorm is carried out.

Complete `main1D.m`. You may use the LehrFEM routines for 1D finite elements. To take into account the Dirichlet boundary conditions, use the function `assemDir_1D.m` provided in the handout.

(9.4h) In this and the following subproblem, we consider the cases $\alpha = 1.1,\ 1.2,\ 1.3,\ 1.4,\ 1.5$. Through numerical experimentation, we are interested in finding, for each α, the minimal mesh grading factor $\bar{\beta} > 0$ which gives the optimal convergence rate in the H^1-seminorm.
For each α, plot the rates as functions of β. What kind of behavior do you observe?

HINT: For each value of α loop over $\beta \in [1, 3]$ with sufficiently small stride. Then compute the finite element solution for these combinations of α and β for $M = 10 \cdot 2^l$, $l = 2, \ldots, 8$, determine the H^1-seminorm of the discretization errors and estimate the rate of convergence in terms of the problem size $N := M - 1$. Plot these rates as functions of β.

(9.4i) Based on what you observed in subproblem (9.4h), plot and try to guess a formula for $\bar{\beta}$ depending on α.

HINT: To find $\bar{\beta}$, the minimal mesh grading factor which gives the optimal convergence rate in the H^1-seminorm, consider a fine partition for the range of β (e.g. subdivide the interval $[1, 3]$ in 50 points).
Furthermore, consider that a very small increment in the convergence rate (e.g. 1% as relative increment) can be considered negligible.

Listing 9.7: Testcalls for Problem 9.4

```
A = STIMA_Lapl_1D([0,0.1])
[U,rate] = main1D(1.1,2);
rate
```

Listing 9.8: Output for Testcalls for Problem 9.4

```
>> test_call

A =

    10   -10
   -10    10

rate =

    0.9736
```

Published on April 29.

To be submitted on May 6. **MATLAB:** Submit all files in the online system. Include the files that generate the plots. Label all your plots. Include commands to run your functions. Comment on your results.

References

[NPDE] Lecture Slides for the course "Numerical Methods for Partial Differential Equations", SVN revision # 54491.

[NCSE] Lecture Slides for the course "Numerical Methods for CSE".

[LehrFEM] LehrFEM manual.

Last modified on May 1, 2013