R. Hiptmair
L. Scarabosio
C. Urzua Torres

Spring Term 2015

Numerical Methods for Partial Differential Equations

ETH Zürich
D-MATH

Homework Problem Sheet 12

Problem 12.1 Radau-3 Timestepping (Core problem)

This short problem studies the full discretization of a linear second-order parabolic initial-boundary value problem as discussed in [NPDE, Section 6.1.1]. The focus will be on the implementation of higher-order implicit Runge-Kutta timestepping.

We consider the parabolic IBVP with homogeneous Dirichlet boundary conditions and zero initial conditions

$$\begin{aligned} \dot{u} - \Delta u &= f(\boldsymbol{x},t) && \text{in } \Omega \times (0,1), \\ u &= 0 && \text{on } \partial\Omega \times (0,1), \\ u &= 0 && \text{on } \Omega \times \{0\}, \end{aligned} \tag{12.1.1}$$

where the spatial domain Ω is the unit disk

$$\Omega := \{\boldsymbol{x} \in \mathbb{R}^2,\ \|\boldsymbol{x}\| < 1\}.$$

The time-dependent source function is given by

$$f(\boldsymbol{x},t) = \begin{cases} 1 & \text{, if } \left\| \boldsymbol{x} - \frac{1}{2}\begin{pmatrix}\cos \pi t \\ \sin \pi t\end{pmatrix} \right\| < \frac{1}{2}, \\ 0 & \text{elsewhere.} \end{cases}$$

We pursue the method of lines approach, see [NPDE, Section 6.1.4]. Spatial discretization relies on linear Lagrangian finite elements on a triangular mesh $\mathcal{M}$, that is, $V_{0,N} = \mathcal{S}^0_{1,0}(\mathcal{M})$, using a polygonal approximation of $\partial\Omega$.

When higher order timestepping schemes are desired for the initial value problem arising from the method of lines approach, L(π)-stable implicit Runge-Kutta methods ($\rightarrow$ [NPDE, Def. 6.1.39]) are the methods of choice. One example is the Radau-3 scheme, a 2-stage implicit Runge-Kutta method, whose description via a Butcher scheme is given in [NPDE, Eq. (6.1.86)].

(12.1a) Devise and implement (in C++ or MATLAB) a numerical experiment for determining the order of (algebraic) convergence of the Radau-3 method, when applied to the initial value problem for a scalar ODE $\dot{y} = -y$, $y(0) = 1$, on $[0,2]$. Which order do you find?

HINT: [NPDE, § 1.6.26] discussed "measurements" for rates of convergence. Measure the error $\max_k |y(t_k) - y^{(k)}|$ for various timestep sizes $\tau > 0$ (equidistant timesteps).

(12.1b) The application of s-stage Runge-Kutta single step methods according to [NPDE, Def. 6.1.39] entails the computation of s increments in each timestep. For the linear ordinary differential equation [NPDE, Eq. (6.1.29)] arising from the method of lines approach to a linear parabolic evolution problem the increments can be obtained by solving a linear system of equations, see [NPDE, § 6.1.42] and [NPDE, Eq. (6.1.43)].

Describe this linear system of equations (in block form) for the RADAU-3 method for (12.1.1) using the notations $\mathbf{A}$ and $\mathbf{M}$ for the $N \times N$ finite element Galerkin matrices associated with the bilinear form for $-\Delta$ and the $L^2(\Omega)$-inner product, respectively. Write $\vec{\varphi}(t)$ for the time-dependent right hand side vector obtained by Galerkin discretization of the source term.

(12.1c) Complete the class Radau3 provided in the `svn` repository, by implementing the method

```
void computeSystemMatrix_(Matrix & MK, Matrix & A, double dt)
```

which takes as argument the timestep `dt`, computes the required triplets, and finally fills the coefficient matrix `MK`$\in \mathbb{R}^{2N,2N}$ for the linear system supplying the increments, see subproblem (12.1b). The argument `A` returns the stiffness Matrix $\mathbf{A} \in \mathbb{R}^{N,N}$, $N := \dim \mathcal{S}_1^0(\mathcal{M})$, for $-\Delta$.

HINT: First get the triplets for the mass matrix and `S`, then build the system block matrix `MK` on the triplets level by using an offset. The use of index-value triplets for the initialization of sparse matrices in the C++ template library Eigen is explained in [NPDE, Rem. 3.6.45]. [NPDE, Code 3.6.49] provides an example for the use of triplets.

HINT: As usual, you are required to use your previous implementations of `DofHandler`, `MatrixAssembler`, `VectorAssembler`, `BoundaryDofs`, and local assemblers, developed in subproblems 7.4, 8.1, and 8.2 (also available in the corresponding solution folders). You will find the new required files for this problem in

```
assignments_codes/assignment12/Problem1
```

(12.1d) Now, inside the class Radau3, implement the method

```
template <class Function>
void solve(Function const& f, double dt);
```

for solving (12.1.1) approximately based on the spatial and temporal discretizations as described above. The argument `f` gives the load function in procedural form (where $f = f(x,t)$), and `dt` specifies the (fixed) timestep.

HINT: Remember `LocalFunction` takes as an argument a load function depending only on `x`. You may use a `lambda` function to convert `f` to a function depending only on `x` (for a fixed `t`).

(12.1e) Complete `main.cc` To solve and plot the approximate solution of (12.1.1) at final time $T = 1$ obtained by the method from subproblem (12.1c) using the mesh `circle_320.msh` and the timestep $\tau = 0.02$. Plot the obtained solution using `Paraview`.

HINT: For validation purposes you may compare your plot with that provided on the lecture's `svn` repository.

Problem 12.2 Decaying Solution by Implicit Euler Timestepping (Core problem)

In this problem we practice the diagonalization technique that was a key tool in the stability analysis of single step methods for semi-discrete parabolic evolution problems, see the presentation in [NPDE, Section 6.1.5.2] and [NPDE, Eq. (6.1.61)].

In class we learned that solutions of the abstract variational linear parabolic evolution problem

$$\begin{aligned} \mathsf{m}(\dot{u},v) + \mathsf{a}(u,v) = 0 \quad \forall v \in V_0, \quad 0 < t < T, \\ u(0) = u_0 \in V_0, \end{aligned} \tag{12.2.1}$$

where both m and a are symmetric positive definite bilinear forms on V_0 that satisfy, see [NPDE, Eq. (6.1.20)],

$$\exists \gamma > 0 : \quad \mathsf{a}(v,v) \geq \gamma \mathsf{m}(v,v) \quad \forall v \in V_0, \tag{12.2.2}$$

display an exponential decay of their m-norm and energy norm, see [NPDE, Lemma 6.1.22].

A simple L(π)-stable [NPDE, Def. 6.1.84] implicit timestepping scheme for the semi-discrete linear parabolic evolution problem

$$\begin{aligned} \mathbf{M}\left\{\frac{\mathrm{d}}{\mathrm{d}t}\vec{\mu}(t)\right\} + \mathbf{A}\vec{\mu} = 0, \quad 0 < t < T, \\ \vec{\mu}(0) = \vec{\mu}_0, \end{aligned} \tag{12.2.3}$$

arising from the *method of lines* ($\rightarrow$ [NPDE, Section 6.1.4]) is the implicit Euler method, see [NPDE, Eq. (6.1.36)]. Here, $\mathbf{M}$ and $\mathbf{A}$ denote the Galerkin matrices ($\rightarrow$ [NPDE, Section 3.2]) associated with m and a w.r.t. an ordered basis of a finite-dimensional trial and test space $V_{0,N} \subset V_0$.

The question is, to what extent the sequence $\vec{\mu}^{(j)}$ generated by the implicit Euler method inherits the exponential decay of the norms stated in [NPDE, Lemma 6.1.22].

(12.2a) Write down the formula for a step of the implicit Euler method producing $\vec{\mu}^{(j)}$ from $\vec{\mu}^{(j-1)}$. Write $\tau > 0$ for the size of the timestep.

(12.2b) From the generalized eigenvalue problem $\mathbf{A}\vec{\mu} = \lambda\mathbf{M}\vec{\mu}$ we deduced the existence of a regular matrix $\mathbf{T} \in \mathbb{R}^{N\times N}$ such that (*cf.* the diagonalization technique discussed in [NPDE, Section 6.1.5.2], and, in particular [NPDE, Eq. (6.1.58)], [NPDE, Suppl. 6.1.59])

$$\mathbf{AT} = \mathbf{MTD}, \quad \mathbf{D} := \operatorname{diag}(\lambda_1, \ldots, \lambda_N), \quad \mathbf{T}^\top\mathbf{MT} = \mathbf{I}.$$

Rewrite the implicit Euler step from subproblem (12.2a) in terms of the transformed coefficient vectors $\vec{\eta}^{(k)} := \mathbf{T}^\top\mathbf{M}\vec{\mu}^{(k)}$, $k = j-1, j$.

(12.2c) What will the norms $\left\|\vec{\xi}\right\|_{\mathbf{M}} := (\vec{\xi}^\top\mathbf{M}\vec{\xi})^{\frac{1}{2}}$ and $\left\|\vec{\xi}\right\|_{\mathbf{A}} := (\vec{\xi}^\top\mathbf{A}\vec{\xi})^{\frac{1}{2}}$ of a coefficient vector $\vec{\mu}$ become for the transformed vector $\vec{\eta} := \mathbf{T}^\top\mathbf{M}\vec{\mu}$?

(12.2d) Express $\mathsf{m}(u_N, u_N)$ and $\mathsf{a}(u_N, u_N)$ through the coefficient vector $\vec{\mu}$ associated with a function $u_N \in V_{0,N}$ from the discrete Galerkin trial/test space and through the Galerkin matrices $\mathbf{A}$ and $\mathbf{M}$. How does (12.2.2) read when stated in terms of matrices and coefficient vectors? What is the relationship of γ and the generalized eigenvalues λ_i?

(12.2e) In the sequel we assume a uniform timestep $\tau > 0$. Show that

$$\left\|\vec{\boldsymbol{\mu}}^{(j)}\right\|_{\mathbf{M}} \leq \frac{1}{1+\gamma\tau}\left\|\vec{\boldsymbol{\mu}}^{(j-1)}\right\|_{\mathbf{M}}, \tag{12.2.4}$$

where $\gamma > 0$ is the constant from (12.2.2).

HINT: There are two ways to tackle the problem: rephrase (12.2.4) in terms of $\vec{\boldsymbol{\eta}}^{(k)}$ and look at the implicit Euler method for these transformed coefficient vectors, see subproblem (12.2b). This is another application of the diagonalization technique.

Alternatively, you may use the Cauchy-Schwarz inequality

$$\vec{\boldsymbol{\xi}}^{\top}\mathbf{M}\vec{\boldsymbol{\zeta}} \leq \left\|\vec{\boldsymbol{\xi}}\right\|_{\mathbf{M}}\left\|\vec{\boldsymbol{\zeta}}\right\|_{\mathbf{M}}, \tag{12.2.5}$$

and the implicit Euler recursion formula from subproblem (12.2a)

(12.2f) Now show (12.2.4) with $\|\cdot\|_{\mathbf{M}}$ replaced with $\|\cdot\|_{\mathbf{A}}$.

HINT: Here it is recommended to use diagonalization and the result of subproblem (12.2c).

(12.2g) What is the relationship of the estimates obtained in subproblems (12.2e) and (12.2f) with [NPDE, Lemma 6.1.22].

HINT: Bound $\left\|\vec{\boldsymbol{\mu}}^{(j)}\right\|_{\mathbf{M}}$ and $\left\|\vec{\boldsymbol{\mu}}^{(j)}\right\|_{\mathbf{A}}$ in terms of $\left\|\vec{\boldsymbol{\mu}}^{(0)}\right\|_{\mathbf{M}}$ and $\left\|\vec{\boldsymbol{\mu}}^{(0)}\right\|_{\mathbf{A}}$, respectively. Then, for fixed t consider $\tau \to 0$ and $j \approx t/\tau \to \infty$. Remember the limit

$$\mathrm{e}^{t} = \lim_{n\to\infty}(1+\tfrac{t}{n})^{n}\,, \quad t \in \mathbb{R}\,.$$

Problem 12.3 Radiative Cooling

This problem is dedicated to the full spatial and temporal discretization of a 2nd-order parabolic evolution problem, see [NPDE, Section 6.1]. It will also ask for implementation in C++ in later sub-problems.

The evolution of the temperature distribution $u = u(\boldsymbol{x}, t)$ in a homogeneous "2D body" (occupying the space $\Omega \subset \mathbb{R}^2$) with convective cooling ($\to$ [NPDE, Ex. 2.7.5]) is modelled by the linear second-order parabolic initial-boundary value problem (IBVP) with flux (spatial) boundary conditions ($\to$ [NPDE, § 6.1.9])

$$\begin{aligned} \frac{\partial u}{\partial t} - \Delta u &= 0 && \text{in } \Omega \times [0,T], \\ -\mathbf{grad}\, u \cdot n &= cu && \text{on } \partial\Omega \times [0,T], \\ u(\boldsymbol{x},0) &= u_0(\boldsymbol{x}) && \text{in } \Omega, \end{aligned} \tag{12.3.1}$$

with $c > 0$.

We pursue a method of lines approach, see [NPDE, Section 6.1.4]. For the spatial Galerkin semi-discretization of (12.3.1) we employ linear finite elements on a triangular mesh $\mathcal{M}$ of Ω (FE space $\mathcal{S}_1^0(\mathcal{M})$) with polygonal boundary approximation.

(12.3a) Derive the spatial variational formulation of the form $\mathsf{m}(\dot{u}, v) + \mathsf{a}(u, v) = \ell(v)$ for (12.3.1), with suitable bilinear forms a and m, and linear form ℓ. Do not forget to specify the function spaces for $u(t,\cdot)$ and the test function v.

HINT: Combine the considerations leading to [NPDE, Eq. (6.1.14)] with the approach explained in [NPDE, Ex. 2.9.6].

(12.3b) Argue why the total thermal energy

$$E(t) := \int_\Omega u(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x}\;,$$

decreases with time, if $u_0(\boldsymbol{x}) > 0$ for all $\boldsymbol{x} \in \Omega$.

HINT: Appeal to the heat conduction background to justify the assumption that $u(\boldsymbol{x},t) \geq 0$ for all $(\boldsymbol{x},t)$. Use test function $v \equiv 1$ in the variational formulation.

(12.3c) Compute the local mass matrix $\mathbf{M}_{\widehat{K}}$ corresponding to $\mathsf{m}(\cdot,\cdot)$ and the local stiffness matrix $\mathbf{A}_{\widehat{K}}$ corresponding to $\mathsf{a}(\cdot,\cdot)$ for the unit triangle $\widehat{K}$ with vertices $\binom{0}{0}$, $\binom{1}{0}$ and $\binom{0}{1}$. Assume that the edge connecting $\binom{0}{0}$ and $\binom{1}{0}$ forms part of $\partial\Omega$ and that the coefficient c is constant along this edge.

HINT: See [NPDE, Def. 3.6.35] for the definition of local matrices, and [NPDE, Section 3.3.5] for concrete formulas. [NPDE, Lemma 3.6.61] for $d = 2$ may also come handy.

(12.3d) Now we turn to the full spatial semi-discretization of (12.3.1). Template files for the new classes you will need to write are available in the lecture `svn` repository

`assignments_codes/assignment12/Problem3`

In `EvSolver.hpp`, implement the method

```
template <class Function>
computeTriplets_(Function const& c, Triplet &tripA, Triplet &tripM)
```

that computes the triplet vectors tripM and tripA for $\mathbf{M},\mathbf{A} \in \mathbb{R}^{N\times N}$, $N := \dim \mathcal{S}_1^0(\mathcal{M})$, for the semi-discrete evolution

$$\mathbf{M}\frac{\mathrm{d}}{\mathrm{d}t}\vec{\boldsymbol{\mu}}(t) + \mathbf{A}\vec{\boldsymbol{\mu}}(t) = 0 \tag{12.3.2}$$

resulting from the $\mathcal{S}_1^0(\mathcal{M})$-based finite element semi-discretization of (12.3.1), when standard nodal bassis functions are used. Here, the argument `c` supplies the value of c.

HINT: The use of index-value triplets for the initialization of sparse matrices in the C++ template library Eigen is explained in [NPDE, Rem. 3.6.45]. [NPDE, Code 3.6.49] provides an example for the use of triplets.

HINT: Do not use the result of sub-problem (12.3c) (which is only valid for a very special triangle). Use your already implemented `DofHandler`, `MatrixAssembler`, `VectorAssembler`, `BoundaryDofs`, and local assemblers, developed in subproblems 7.4, 8.1, and 8.2 (also available in the corresponding solution folders).

HINT: Since we are dealing with Robin B.C., your bilinear form should have a term which is integrating over the boundary. This means the local assembler has to take edges as element type instead of triangles, reason why you cannot use your analytical implementations. This was already done in subproblem (8.2g). Solution to subproblem (8.1e) might also be useful.

(12.3e) In [NPDE, Ex. 6.1.85] you learned about the L-stable SDIRK-2 implicit 2-stage Runge-Kutta method described by the Butcher scheme

$$\begin{array}{c|cc} \lambda & \lambda & 0 \\ 1 & 1-\lambda & \lambda \\ \hline & 1-\lambda & \lambda \end{array} \qquad \lambda := 1 - \tfrac{1}{2}\sqrt{2}\,. \tag{12.3.3}$$

Derive the recursion obtained by applying the SDIRK-2 method to the linear scalar ODE $\dot{y} = -\gamma y$, $\gamma \in \mathbb{R}$.

(12.3f) Determine the order of the SDIRK-2 single step method empirically by applying it to the initial value problem $\dot{y} = -y$ on $[0,2]$, $y(0) = 1$. To that end write a short code in MATLAB or C++.

(12.3g) Give a rigorous proof that the SDIRK-2 method is $\mathrm{L}(\pi)$-stable ($\rightarrow$ [NPDE, Def. 6.1.84]).

HINT: Of course, the recursion found in subproblem (12.3e) has to be used. Then the problem boils down to discussing the behavior of a rational function on the (positive) real axis.

(12.3h) [NPDE, § 6.1.42] presents the linear system of equations for the increments of an s-stage Runge-Kutta method when applied to the semi-discrete evolution (12.3.2). State this linear system explicitly in block form for the special case of the SDIRK-2 method.

(12.3i) Write a method

```
template <class Function>
void solve(Vector & U, Function const& c);
```

in EvSolver that carries out m uniform timesteps of the in order to solve (12.3.1) over the time interval $[0,1]$. The finite element Galerkin discretization from subproblem (12.3d) is used in space. The argument `U` is a column vector that passes the values of the initial temperature distribution in the vertices of the mesh. The return value provides the basis coefficients of the approximation of $u(\cdot,1)$ of u at $t = 1$. This function will be called within the main.

HINT: From [NPDE, Def. 6.1.39] and [NPDE, Eq. (6.1.41)] it should be clear how to obtain the linear systems of equations [NPDE, Eq. (6.1.43)], [NPDE, Eq. (6.1.44)] for the Runge-Kutta increments.

(12.3j) In EvSolver, implement a method

```
double average_(Vector const& U);
```

that computes $\int_\Omega u \,\mathrm{d}\boldsymbol{x}$ for $u \in \mathcal{S}_1^0(\mathcal{M})$. The argument `U` passes the coefficients of u w.r.t. the standard nodal basis of $\mathcal{S}_1^0(\mathcal{M})$.

HINT: Your implementation for subproblem (7.3a) might be useful.

(12.3k) For the evolution problem (12.3.1) on $\Omega = (0,1)^2$ track the behavior of the thermal energy

$$E(t) = \int_\Omega u(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x} \tag{12.3.4}$$

over the period $[0, T]$ for $u_0 \equiv 1$, $\gamma = 1$. Use the fully discrete evolution implemented in EvSolver::solve() and extend it, so it also computes approximations for $E(t_k)$ for $k = 0, \ldots, m$ (t_k are the points of the equidistant temporal grid). Then implement a method Vector getE() which returns a Vector E containing said approximations.

Complete `main.cc` to compute $u(x, t)$ for $t = 1$ for the mesh supplied in the file `square_32.msh`. Make a plot of u for $t = 0$ and $t = 1$ using `Paraview`. Plot the approximation for $E(t)$ that you have computed as a function of t for $m = 100$.

HINT: The plot for E is depicted in Figure 12.1.

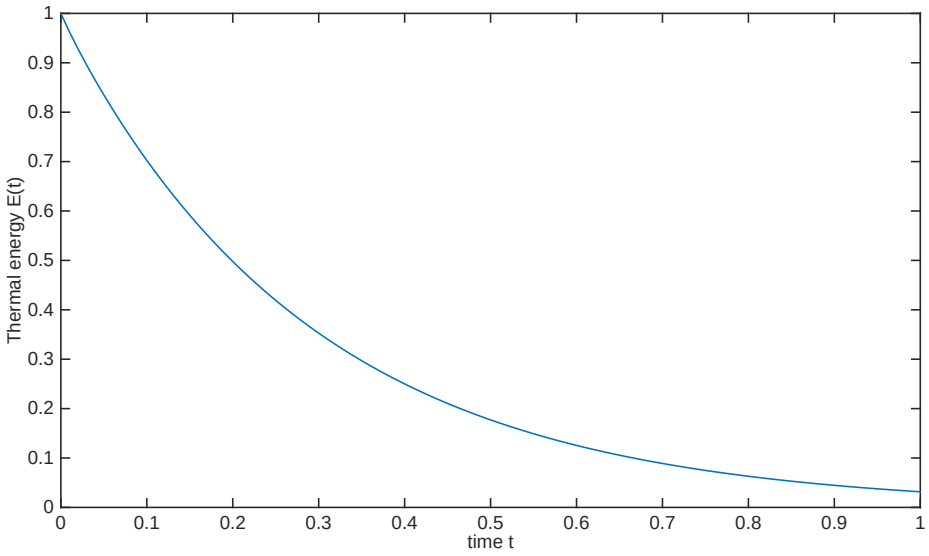

Figure 12.1: Result of subproblem (12.3k)

Published on 13.05.2015.

To be submitted on 20.05.2015.

References

[NPDE] Lecture Slides for the course "Numerical Methods for Partial Differential Equations".SVN revision # 76119.

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

Last modified on May 13, 2015