Prof. Dr. B. Haasdonk Exercise for the Lecture 25.1.2023

Introduction to the Numerics of PDEs

WS 2022/2023– Programming Sheet 3

Submission: until 8.2.2023 at 11:30 in ILIAS.

Discussion: at 10.2.2023 in the exercise session.

Solve the following programming tasks by implementing corresponding programs. Insert your name and student-identification number in the head of your programs. Generate a PDF file with your solution (name, student-id, diagrams, outputs, explanations, insights) Upload the PDF file and your executable programs to ILIAS before the submission deadline.

The goal of this sheet is to implement and analyze an approximation of the solution of the heat equation

$$\begin{aligned} \partial u(x,t) - \Delta u(x,t) &= f(x,t), \quad (x,t) \in \Omega \times (0,T) \\ u(x,t) &= 0, \quad x \in \partial\Omega \times (0,T) \\ u(x,0) &= u_0(x), \quad x \in \Omega, \end{aligned}$$

with $\Omega = (0,1)^2, T = 1$. We test the implementation with

$$\begin{aligned} f(x,t) &:= \sin(\pi x_1)\sin(2\pi x_2)(-5\pi\sin(5\pi t) + 5\pi^2\cos(5\pi t)) \\ u_0(x) &:= \sin(\pi x_1)\sin(2\pi x_2) \end{aligned}$$

which are generated with the exact solution

$$u(x,t) := \sin(\pi x_1)\sin(2\pi x_2)\cos(5\pi t).$$

For the following task you may use a solution of FEM assembling routines from Programming Sheet 2.

Programming Task (θ-scheme) ....

We want to solve the following system of ODEs

$$\begin{aligned} Mv'(t) + Av(t) &= r(t) \\ v(0) &= v_0, \end{aligned}$$

where the positive definite matrices $M, A \in \mathbb{R}^{n\times n}$, the initial vector $v_0 \in \mathbb{R}^n$ and the function $r : [0,T] \to \mathbb{R}^n$ are given. For a time-step width $\Delta t > 0$ we define a time discretization $0 =: t^0 < t^1 < \ldots < t^K := T$ with $t^k - t^{k-1} = \Delta t$ for all $k = 1 \ldots K$ and we assume that $T/\Delta t \in \mathbb{N}$. For a given $\theta \in [0,1]$ the θ-scheme approximates v by constructing approximate solutions $v^k \approx v(t^k)$ on the time grid. This approximation is obtained by setting $v^0 := v_0$ and then for $k = 0, \ldots K-1$ computing v^{k+1} iteratively by solving the system

$$(M + \theta\Delta tA)v^{k+1} = (M - (1-\theta)\Delta tA)v^k + \Delta t(1-\theta)\underbrace{r(\overset{k\cdot\Delta t}{t^k})}_{\text{safe one func. eval}} + \Delta t\theta r(\overset{(k+1)\Delta t}{t^{k+1}}).$$

for loop? can we avoid it?

Write a function

```
v = theta_method(M, A, v0, r, T, K, theta)
```

that implements this method. The variable `r` should be a function that can be evaluated in a scalar time point $t \in \mathbb{R}$ and returns a vector. The output of the routine should be a $n \times (K+1)$ matrix consisting of the vectors v^k for $k = 0, \ldots, K$ as columns.

Programming Task (Space Discretization) . . . .

We want to use the triangulation $\mathcal{T}_h$ of Ω and the corresponding space of piecewise linear and globally continuous functions $\mathbb{P}_{1,0}(\mathcal{T}_h)$ with nodal basis functions $\{\varphi_i\}_{i=1}^{m_{1,0}}$ corresponding to the interior nodes $\{x_i\}_{i=1}^{m_{1,0}}$. We assemble the matrices $M_h, A_h \in \mathbb{R}^{m_{1,0} \times m_{1,0}}$ and vectors $u_h^0, f_h(t)$ defined by

$$
\begin{aligned}
(M_h)_{i,j} &:= \langle \varphi_i, \varphi_j \rangle_{L^2(\Omega)} \\
(A_h)_{i,j} &:= \langle \nabla\varphi_i, \nabla\varphi_j \rangle_{L^2(\Omega)^2} \\
(u_h^0)_i &:= u_0(x_i) \\
(f_h(t))_i &:= {\scriptstyle f_h(t)} \int_\Omega \overset{f_2(x)}{f(x,t)}\varphi(x)dx, \qquad \text{for all } t \in [0,T].
\end{aligned}
$$

Write a function

$$[\texttt{A\_h}, \texttt{M\_h}, \texttt{u\_h0}, \texttt{f\_h}] = \texttt{space\_discretization}(\ldots)$$

that computes the (sparse) matrices and the vectors. The vector `f_h(t)` should be a function, that can be evaluated in a scalar time t and returns a vector `f_h(t)`. To obtain this, you can assume separability of the PDE right hand side f in the sense that

$$f(x,t) = f_1(t)f_2(x),$$

and then

$$\int_\Omega f(x,t)\varphi(x)dx = f_1(t)\int_\Omega f_2(x)\varphi(x)dx.$$

Thus, the integral can be computed by a quadrature rule and the result can then be multiplied by the function $f_1(t)$.

Programming Task (Test and Error Analysis) . . . .

Write a function

$$\texttt{u\_h} = \texttt{discretization\_heat\_problem}(\ldots)$$

that approximates the solution of the heat equation. The solution `u_h` is the result of the θ-scheme with $A = A_h, M = M_h, v^0 = u_h^0, r = f_h$. Use this solver for the following numerical investigations

Prog 2 → local error?

a) Choose an appropriate error measure and compute the error with respect to the exact solution for $\theta = 0, 1/2, 1$, fixed mesh and for varying time step widths $\Delta t > 0$. What do you observe with respect accuracy and stability?

b) For fixed $\theta = 0, 1/2, 1$ compute and plot the error for simultaneous decreasing values of Δt and mesh width h. What is the observed order of convergence?