Programming project – Numerics for Instationary Differential Equations

Part I - Time integration

Implement the Radau5-method (Radau IIA of order 5) and the BDF2 method with constant step size for a linear matrix ordinary differential equations (ODE).

1) The Radau IIA method is given by the Butcher tableau

$$\begin{array}{c|ccc} \frac{2}{5}-\frac{\sqrt{6}}{10} & \frac{11}{45}-\frac{7\sqrt{6}}{360} & \frac{37}{225}-\frac{169\sqrt{6}}{1800} & -\frac{2}{225}+\frac{\sqrt{6}}{75} \\ \frac{2}{5}+\frac{\sqrt{6}}{10} & \frac{37}{225}+\frac{169\sqrt{6}}{1800} & \frac{11}{45}+\frac{7\sqrt{6}}{360} & -\frac{2}{225}-\frac{\sqrt{6}}{75} \\ 1 & \frac{4}{9}-\frac{\sqrt{6}}{36} & \frac{4}{9}+\frac{\sqrt{6}}{36} & \frac{1}{9} \\ \hline & \frac{4}{9}-\frac{\sqrt{6}}{36} & \frac{4}{9}+\frac{\sqrt{6}}{36} & \frac{1}{9} \end{array}$$

2) The BDF 2 method is given by

$$\frac{3}{2}y_{n+2} - 2y_{n+1} + \frac{1}{2}y_n = hf(t_{n+2}, y_{n+2}).$$

Voluntary task: If you are motivated, you can implement the methods above for general ODEs. Then you further need to solve the appearing nonlinear system with the Newton method. For this consider the lecture notes (page 15 & 16) together with Exercise 9 for a stopping criteria.

Part II - Full discretization of heat equation

Consider the heat equation

$$\begin{cases} \frac{\partial u}{\partial t}(x,t) - \Delta u(x,t) = f(x,t) & \text{in } \Omega \times [0,T], \\ u(x,t) = 0 & \text{on } \Gamma \times [0,T], \\ u(x,0) = u_0(x) & \text{in } \Omega, \end{cases}$$

where Ω is the unit circle. Implement linear finite elements and use the Radau IIA and BDF2 from Part I with constant time step size. We choose $u(x,t) = e^{-t}(1 - \|x\|^2)$ as exact solution and thus $f(x,t) = -e^{-t}(1 - \|x\|^2) + 4e^{-t}$.

Groundwork for this exercise is available on the web page. The second initial value is computed with the exact solution. The load vector $\mathbf{b}(t)$ can be approximated by

$$\mathbf{b}(t)_j = \int_\Omega f(x,t)\varphi_j \, dx \approx \int_\Omega I_h f(x,t)\varphi_j \, dx = \big(\mathbf{M}\mathbf{f}(t)\big)_j,$$

where $\mathbf{f}(t)$ is the vector of nodal values.

Compute solutions using the grid sizes and time step sizes given in the pseudo code. Generate four plots for each time integration method: The L^2-errors and H^1-errors of the approximation to $u(x, 1)$, plotted against h and τ, respectively. The errors are given by

$$\sqrt{\mathbf{e}^T \mathbf{M} \mathbf{e}} \qquad \text{and} \qquad \sqrt{\mathbf{e}^T \mathbf{A} \mathbf{e}},$$

respectively.

Explain how the plots correspond to the convergence theory of parabolic equations.

What is the computational bulk of the whole method?

Hints:

- The grid size given as input parameter to distmesh is not the real grid size of the resulting grid, so you have to compute the real grid size after the grid is generated.
- Use sparse matrices!
- You may have to avoid the smallest value of h, depending on your computer's performance and memory (overall computation time may be around 5 minutes).
- Feel free to come to my office if something is not clear.
- There is one more page that has the programming exercise for the elliptic case which might be a good start before tackling this exercise, but it is optional!

Hand in solutions via mail by June 9, 2026 – georgios.vretinaris@uni-tuebingen.de.
Open door policy – Come to my office or send me an email if you have any questions!

Optional Elliptic case to solve, before dealing with the Parabolic one

Solve with the finite elements method the problem

$$\begin{aligned} -d\Delta u + cu &= f \quad \text{in} \quad \Omega, \\ u &= 0 \quad \text{auf} \quad \partial\Omega, \end{aligned}$$

where the domain Ω and the parameters $d > 0, c \geq 0$ are given as the following:

- Ω: Direct as a triangulation through the matrices (elements, nodes);
- $\partial\Omega$: Through the list of boundary nodes (boundary)
- f: As a function (`func_f.m` or `func_f.jl`).

Implement a function `matrix_assembly(Elements,Nodes)` for the computation of the stiffness- and mass-matrix. Use the exercises 13, 14 and 15. The vector b can also be approximated (in first-order) by

$$b|_j = \int_\Omega f\phi_j \approx \int_\Omega I_h f \phi_j = (Mf)_j$$

An example code for handling the txt-files can be found under: `https://na.uni-tuebingen.de/ex/num4_ss26/Programming_project.zip`.

(a) Solve by first writing your `matrix_assembly` function to construct the linear equation system, which corresponds to the problem above and Ω being the unit circle. A reference solution (`func_solution`) and the inhomogeneous function (`func_f`) are given as a m/jl-file.

Compute the error of the numerical solution for different grids. The triangulations for the unit circle are encoded in the txt-files

`Elements_j.txt`
`Nodes_j.txt` $\qquad j = 1, 2, 3, 4.$

(b) Compute the error in the L^2 norm and the H^1 semi-norm through:

$$\begin{aligned} \|e_h\|^2_{L^2(\Omega)} &= \|\mathbf{e}\|^2_M = \mathbf{e}^T M \mathbf{e}, \\ \|\nabla e_h\|^2_{L^2(\Omega)} &= \|\mathbf{e}\|^2_A = \mathbf{e}^T A \mathbf{e} \end{aligned}$$

and plot the errors (loglog-plot!).