Technische Universität Berlin
Institut für Mathematik
Dr. M. Weiser, J. Ullrich, B. Höveler

WiSe 2022/23
https://isis.tu-berlin.de/course/view.php?id=30963

Numerical Mathematics II for Engineers

Homework 3

Deadline: Submit theoretical and programming exercises on ISIS until **Friday 23:59, November 11th, 2022**.

1. Exercise: Neumann boundary conditions **4 points**

Let A_h denote the matrix arising from the symmetric boundary discretization of Neumann boundary conditions in 1D. Prove that for the eigenvalues λ of this matrix one has at least

$$\mathrm{Real}\lambda \geq 0.$$

Hint: Let A be a square (complex) matrix. Around every element a_{ii} on the diagonal of the matrix, we draw a circle with radius r_i the sum of the norms of the other elements on the same row $r_i = \sum_{j=1, j\neq i}^{n} |a_{ij}|$. Such circles are called Gershgorin discs defined as $C_i := \{c \in \mathbb{C} : |c - a_{ii}| \leq r_i\}$.

Theorem of Gershgorin: Every eigenvalue $\lambda \in \mathbb{C}$ of A lies in one of these Gershgorin discs C_i. The spectrum (i.e. the set of all eigenvalues) of A is contained in the union of all discs $C = \bigcup_{i=1}^{n} C_i$.

Bonus +1: Does this statement hold for the discretization matrix in 2D, too?

2. Exercise: Kronecker product **6 points**

Let $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{p\times q}$, $C \in \mathbb{R}^{n\times r}$, $D \in \mathbb{R}^{q\times s}$. Show the following properties of the Kronecker product:

a) $(A \otimes B)(C \otimes D) = (AC) \otimes (BD) \in \mathbb{R}^{mp\times rs}$,

b) $(A \otimes B)^{\mathsf{T}} = A^{\mathsf{T}} \otimes B^{\mathsf{T}}$,

c) Consider the special case that $m = n$ and $p = q$. Moreover, let λ be an eigenvalue of A with eigenvector x and μ be an eigenvalue of B with eigenvector y. Show that $\lambda\mu$ is an eigenvalue of $A \otimes B$ with eigenvector $x \otimes y$.

Please turn the page!

3. Programming exercise: Neumann boundary conditions for 1D Poisson **2 points**

In exercise 4 on sheet 2, we asked you to solve the 1D Poisson equation

$$-\Delta u = f \quad \text{on } \Omega$$

with $\Omega =]0,1[, u : \mathbb{R} \to \mathbb{R}$ by FDM by using homogeneous Dirichlet boundary conditions on the whole boundary $\partial\Omega$ and $f = 1$. In this exercise, we ask you to solve this 1D Poisson equation with homogeneous Dirichlet boundary conditions on the left side and homogeneous Neumann boundary conditions on the right side of the boundary. Reuse your program from exercise 4 on sheet 2 and simply redefine the according variables.

4. Programming exercise: Singularly perturbed BVP **8 points**

Consider the two-point boundary value problem

$$\begin{cases} -\varepsilon u''(x) + u'(x) = 1, & x \in (0,1), \\ u(0) = u(1) = 0, \end{cases} \tag{1}$$

with small but positive values for ε. Simply setting $\varepsilon = 0$ does not help, because the solution (if it exists) is not close to the one for small values for $\varepsilon > 0$. It also changes the order of the BVP. This behaviour is typical for *singularly perturbed problems*.

The exact solution to (1) is given by

$$u_\varepsilon(x) = x - \frac{e^{-(1-x)/\varepsilon} - e^{-1/\varepsilon}}{1 - e^{-1/\varepsilon}}, \quad x \in (0,1)$$

and shown in the figure below. As we see $u_0(x) = \lim_{\varepsilon \to 0} u_\varepsilon(x) = x$ for $0 \le x < 1$, but $u_0(x)$ does not satisfy the boundary condition at $x = 1$ in a smooth way. Instead, the solution exhibits a thin region near $x = 1$ (also known as boundary layer), where u_ε changes rapidly. The width of the region depends on ε and thereby derivatives of u_ε become large as $x \to 1$ and $\varepsilon \to 0$. That means that the constant $u^{(k)}(\xi)h^{k-2}$ ($k > 2$) in the remainder of the difference quotient for u'' is large. In order to make the remainder small, we need to make h very small.

a) Write a function `uh=a04ex03solve(eps,xh,flag)` which returns the FDM solution of the singularly perturbed problem (1) for given `eps` $= \varepsilon$, grid `xh`, and approximation for the first derivative selected by `flag`. The flag should provide the choice between forward difference operator D^+ (`flag='+'`), central difference operator D^0 (`flag='0'`) and backward difference operator D^- (`flag='-'`). The second derivative is approximated by D^+D^- as calculated in exercise 2 on sheet 2.
Hint: In Matlab, you can use the functions `spdiags` or `gallery('tridiag',...)`.

b) Write a function `[err,uex] = a04ex03error(eps,xh,uh)` that returns the error `err` (measured by the norm $\|\cdot\|_\infty$) between `uh` and the restricted exact solution, which is also to be returned as `uex`.

See next page!

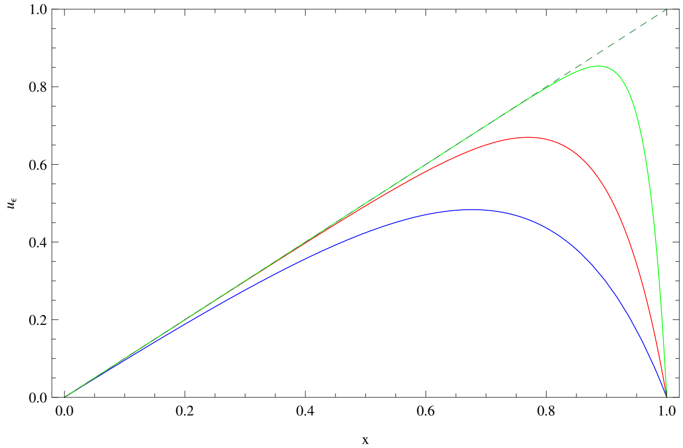

Fig.: u_ε for $\varepsilon \in \{1/5,\ 1/10,\ 1/30\}$ (blue, red, green) and the function $u_0(x) = x$ (dashed)

c) Write a function `xh = a04ex03shishkin(N,sigma)` that generates a column vector of size 2N+1 describing a "Shishkin" grid `xh` that is defined by

$$\texttt{xh(i)} = \begin{cases} \texttt{(i-1)*H}, & \text{for} \quad \texttt{i = 1, ..., N}, \\ \texttt{(1-sigma) + (i-N-1)*h}, & \text{for} \quad \texttt{i = N+1, ..., 2N+1}, \end{cases}$$

where `H = (1-sigma)/N` and `h = sigma/N`.

d) Write a script `a04ex03experiment` that plots the exact and approximated solution for `eps` $= 0.001$ and all $N \in \{5,\ 50,\ 500,\ 5000\}$ on a

i) uniform grid with $h = \frac{1}{2N}$ and forward difference operator D^+,
ii) uniform grid with $h = \frac{1}{2N}$ and central difference operator D^0,
iii) uniform grid with $h = \frac{1}{2N}$ and backward difference operator D^-,
iv) non-uniform Shishkin grid with `N` and `sigma=4*eps*log(2*N)` (`log` is natural logarithm here) and central difference operator D^0.

Use the previously created flags for choosing between the different operators. Create four different figures for i), ii), iii), and iv). Results with different N but same operator should be plotted into the same figure. Use the figure titles to add information about the used operator, the grid, and the error for different N.

total sum: 20 points + 1 bonus

Important notes:

- New homework sheets will usually be made available weekly on Friday after the lecture, on the ISIS webpage. The exercise should be solved in **fixed groups of at most 3 students**, and returned as stated on the assignment.

Please turn the page!

- We recommend using MATLAB, Octave or Python/Numpy to solve the programming exercises. Templates for the solutions, in the form of MATLAB script/function files, or jupyter notebooks are available on the ISIS webpage.
- It is possible to use different programming languages, as long as you agree it with the tutors beforehand.
- Always put the name from every member of your group on the first page of your submission.
- **Please upload the programming exercises in separate files (`.m`, `.py`, `.ipynb`, etc...). Embedding the code as images inside a pdf is not sufficient!**