



Homework Assignment 2

Please submit the following files as indicated below: source code PDF file image file video file

Question 1 | 3 marks | Let $\Omega \subset \mathbb{R}^2$ be a two-dimensional domain, $a : \Omega \rightarrow \mathbb{R}^2$ a two-dimensional, continuously differentiable vector field, $D > 0$ a constant and $g : \partial\Omega \rightarrow \mathbb{R}$ a continuous function.

The steady advection-diffusion problem

$$\begin{aligned} \operatorname{div}(ua) - \operatorname{div}(D\nabla u) &= 0 && \text{in } \Omega \\ u &= g && \text{on } \partial\Omega \end{aligned}$$

describes how a certain density u is transported through the domain Ω .

(a) Show that any strong solution $u \in C^2(\Omega) \cap C(\bar{\Omega})$ of this equation is bounded by its boundary values

$$\min_{s \in \partial\Omega} g(s) \leq u(x) \leq \max_{s \in \partial\Omega} g(s) \quad \text{for all } x \in \bar{\Omega}$$

provided that $\operatorname{div} a = 0$ (“incompressibility condition”). Show all working and clearly indicate what results (e.g. from the notes) you are using in each step.

This equation is elliptic, since the principal part of the differential operator $-D\Delta$ is just a positive multiple of the Laplacian $-\Delta$.

If $\operatorname{div} a = 0$, the PDE is equivalent to (cf page 6 in the notes)

$$a \cdot \nabla u - D\Delta u = 0 \quad \text{in } \Omega.$$

Hence, this equation satisfies the assumptions of the elliptic maximum principle (Thm 2.1.2). Applied to a solution u and to $-u$, we conclude

$$\max_{x \in \bar{\Omega}} u(x) \leq \max_{s \in \partial\Omega} u(s) \quad \text{and} \quad \min_{x \in \bar{\Omega}} u(x) \geq \min_{s \in \partial\Omega} u(s)$$

and thus for all $x \in \bar{\Omega}$


$$\min_{s \in \partial\Omega} g(s) = \min_{s \in \partial\Omega} u(s) \leq \min_{x \in \bar{\Omega}} u(x) \leq u(x) \leq \max_{x \in \bar{\Omega}} u(x) \leq \max_{s \in \partial\Omega} u(s) = \max_{s \in \partial\Omega} g(s).$$

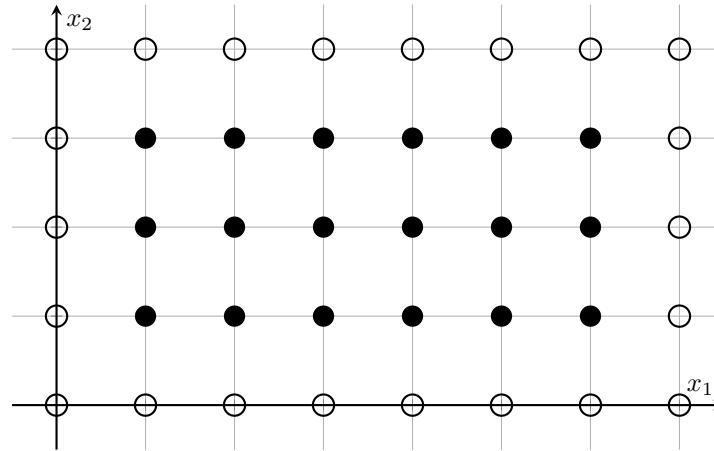
(b) Why do we have to assume $\operatorname{div} a = 0$ to derive these bounds?

Generally, the PDE is equivalent to

$$(\operatorname{div} a)u + a \cdot \nabla u - D\Delta u = 0 \quad \text{in } \Omega,$$

i.e. it contains an order-zero reaction term. The elliptic maximum principle from Thm 2.1.2 is not applicable, unless the coefficient of this term vanishes.

Question 2 | 2 marks |  Today we will implement two more auxiliary functions for the finite difference method on rectangular domains. Initially, $X1$, $X2$, a discretisation F of the source term and an initial guess U for the solution of the PDE are usually given as rectangular arrays. For our computations, we have to re-arrange some of these arrays to column vectors. For plotting and post-processing, we have to re-arrange some column vectors back to rectangular arrays.



If the PDE is equipped with **DIRICHLET** boundary conditions, then the solution values on the boundary (empty circles) are already known. Only the interior solution values (filled circles) are the actual unknowns.

- (a) Write a function `msh2vec` that applies so-called *lexicographical ordering* to re-arrange a rectangular array to a column vector. This function takes the two input variables

U: an array of size $(\text{msh.N}(2) - 1) \times (\text{msh.N}(1) - 1)$ corresponding to function values on the *interior* nodes of the grid only, not the boundary nodes

msh: the output of `meshRectangle`

and returns

u: an array of size $(\text{msh.N}(1) - 1)(\text{msh.N}(2) - 1) \times 1$

The first component of **u** should contain the value of **U** that corresponds to the bottom left interior grid point (filled circle) of the rectangular domain, the second component should correspond to the next grid point to the right etc. Moving from left to right row by row, the last component of **u** will be equal to the value of **U** that corresponds to the interior grid point (filled circle) in the top right corner of the domain.

- (b) Implement a function `U = vec2msh(u,msh)` that undoes the action of `msh2vec`. Write a script `hw2.m` that tests whether for the sample function from `hw1.m` the arrays

$$U = u(\text{msh.X1}(2:\text{end}-1), 2:\text{end}-1), \text{msh.X2}(2:\text{end}-1, 2:\text{end}-1))$$

and

$$V = \text{vec2msh}(\text{msh2vec}(U, \text{msh}), \text{msh})$$

are indeed the same.

Hint: In GNU Octave / MATLAB, the command `reshape` may be helpful.

Update on the submission instructions: You no longer have to include a PDF printout of your code, the source code files are sufficient. Please include all files that are needed to run the program in your submission, even if you have already submitted them for a previous assignment. Thank you!

Your Learning Progress |  What is the one most important thing that you have learnt from this assignment?

What is the most substantial new insight that you have gained from this course this week? Any *aha moment*?
