TMA4212 Num.diff.
Spring 2023

Project 1

Norwegian University of Science and Technology
Department of Mathematical Sciences

Practical information

- *Deadline and hand-in:* Sunday February 26 23:59. Hand in the project in ovsys (pdf file (max 7 pages long) + Jupyter Notebook (should run/compile without errors)).
- *Report:* The report can be written as a pdf-document together with your python code as Jupyter file. More details are present on the wiki page under "Exercises and projects". Write the report as a scientific report, not as a solution to an exercise. Meaning: Describe the problem you want to solve, describe the method you use, write math results as math statements, and make sure there is a connection between theoretical and numerical results etc. Use (readable!) plots when appropriate and explain clearly what you observe and whether the results are expected or not.
- *Grading:* Max 5 points for the report, max 10 points for problem 1, max 5 points for problem 2. Total max score 20 points .
- *Learning objectives:*
  - develop and implement finite difference schemes for elliptic 2-D problems in complex domains;
  - make test problems to test the code, numerically estimate and present graphically convergence rates;
  - perform theoretical error analysis in some cases;
  - identify and solve potential deficiencies of a scheme;
  - communicate the results in a scientific manner.

Some advice:

- *Implementation:* Divide the coding of your solvers into smaller parts, test each part of the code before moving to the next.
- *Writing:* Imagine you are writing for fellow students that have not seen the project description. Try to make them understand and be interested in your results. Writing takes time, so start early and rewrite parts if necessary, that is part of the process.
- *Time organization:* Plan how much time you are willing to invest. Start early. If you are stuck at some point, it could be an idea to drop it and concentrate on writing a good report instead.

Part 1: Heat distribution in anisotropic materials

The Poisson equation is used to model the stationary temperature distribution T of a solid Ω. If the heat conductivity is κ, and the solid has internal heat sources f (an energy density), then conservation of energy, Fourier's law for the heat flux, and stationarity ($\partial_t T = 0$), give the following model

$$-\nabla \cdot (\kappa \nabla T) = f \qquad \text{in} \qquad \Omega. \tag{1}$$

In anisotropic materials the heat flows faster in some directions than others, and this means that the κ is a matrix. We will focus on two dimensional models with two distinguished directions for the heat flow:

$$\vec{d_1} = (1,0) \qquad \text{and} \qquad \vec{d_2} = (1,r) \quad \text{where} \quad r \in \mathbb{R}.$$

After normalisation this gives a heat conductivity of the form

$$\kappa = \begin{pmatrix} a+1 & r \\ r & r^2 \end{pmatrix},$$

where $a > 0$ is a constant and

$$R := \frac{a}{|\vec{d_2}|^2} = \frac{a}{1+r^2}$$

is the relative strength of the conductivity in the $\vec{d_1}$ versus $\vec{d_2}$ direction. In this case

$$\nabla \cdot (\kappa \nabla T) = (a+1)\partial_x^2 u + 2r\partial_x\partial_y u + r^2\partial_y^2 u = a\partial_x^2 u + (\vec{d_2} \cdot \nabla)^2 u. \tag{2}$$

Note in particular the second order directional derivative!

$\boxed{1}$ Let $\Omega = [0,1] \times [0,2]$ with Dirichlet boundary conditions $u = g$ on $\partial\Omega$.

a) Let $r = 2$ and consider a grid with step sizes $h = \frac{1}{M}$ ($M \in \mathbb{N}$) and $k = 2h$ in the x and y directions. Discretise and solve the problem numerically using second order central differences in the directions $\vec{d_1}$ and $\vec{d_2}$. Write the program experimenting with some simple choices of g and f.

OBS: Do NOT use the form with mixed derivatives. Discretised mixed derivatives are unstable. Discretise the 2nd order directional derivatives instead.

b) Show that the scheme in part (a) is monotone. Write down its stencil. Use the discrete max principle and the comparison function $\phi = \frac{1}{2}x(1-x)$ to show L^∞ stability. Derive an error bound. What is the rate of convergence for smooth solutions?

c) Test the scheme, check the convergence rate, plot the solution and log-log plots of the errors in some cases.

Hint: To find an exact solution of the problem, you may modify the right hand side f and the initial data. Then as explained in the lectures, any nice function will be a solution for a suitable choice of f and the initial data.

So far we were lucky since the directional derivatives could be computed on the naive/standard grid. Let us now take a bad direction so that this is no longer possible: Let $r \neq 2$ and irrational. We now test one idea to resolve this new problem.

d) Introduce a new grid with step sizes $h = \frac{1}{M}$ and $k = |r|h$ in the x and y directions. Adapt the scheme of part a) to this grid. Explain that the grid now will miss the upper boundary ($y = 2$) and how to overcome this problem. Implement the scheme and experiment. Test the scheme and check the convergence rate numerically. Plot the solution in one interesting case.

OBS: We ask for no analysis in problem d).

$\boxed{2}$ Let Ω be the domain in the first quadrant enclosed by the parabola $y = 1 - x^2$ and the two axis. The boundary then consists of the curves

$$\gamma_1 = [0, 1] \times \{0\}, \quad \gamma_2 = \{0\} \times [0, 1], \quad \text{and} \quad \gamma_3 = \big\{(x, 1 - x^2) \ : \ x \in [0, 1]\big\}.$$

Figure 1: The domain Ω

Solve numerically the Dirichlet boundary value problem for (1) in the isotropic case when when $\kappa = I$ and

$$\nabla \cdot (\kappa \nabla T) = \Delta T.$$

Implement the scheme and experiment. Test both strategies suggested in the lectures for handling irregular boundaries: (i) modify the discretisation near the boundary, and (ii) fatten the boundary. Which one is easier/faster?

Some hints: You may use the normal or any other extension when programming fattening the boundary. Note that if the boundary condition is constant along γ_3, then all extensions are the same (constant).

For any point $\vec{x}_P = (x_P, y_P)$ in the first quadrant, by elementary geometry or minimising the distance, the normal projection onto the curve γ_3 is

$$\vec{x}_Q = (r, 1 - r^2) \quad \text{where } r \text{ is a positive solution of} \quad x_p + r(1 - 2y_p) - 2r^3 = 0.$$

One idea for the numerical solution is to use the package `sympy` and relative functions (`solve` etc.). If you do not want to solve this equation numerically, you can use the extensions along the directions $(1, 0)$ and $(0, 1)$ in different parts of the domain.