TMA4212: Project 1

Heat distribution in anisotropic materials & in irregular domains

NTNU

Written by
Ole Gunnar Røsholt Hovland
Alexander Edward Hatle
Jan Haakon Melka Trabski

2023

Abstract

The goal of this project was to look at numerical problems and different methods for solving two-dimensional heat distribution in anisotropic and isotropic materials. The first part of the project was to look at a problem where we discretised the heat distribution in an anisotropic material. A difficulty in this problem was to avoid discretization with mixed derivatives - since this is numerically unstable. This can be avoided by redefining the derivative by the use of directional derivatives. We then show monotonicity and stability, derive an error bound, and find the rate of convergence for smooth solutions. The last part of the thesis was to look at an isotropic material with a more complicated boundary. The domain was redefined as the domain in the first quadrant enclosed by the parabola $y = 1 - x^2$ and the two axes. Different boundary methods, such as fattening the boundary and modifying the stencil near the boundary were used to compensate for the new domain.

Contents

1 Introduction

We wish to model the stationary temperature distribution in an antistrophic material. This can be described with the following equation.

$$-\nabla \cdot (\kappa \nabla T) = f \quad \text{in } \Omega \tag{1.1}$$

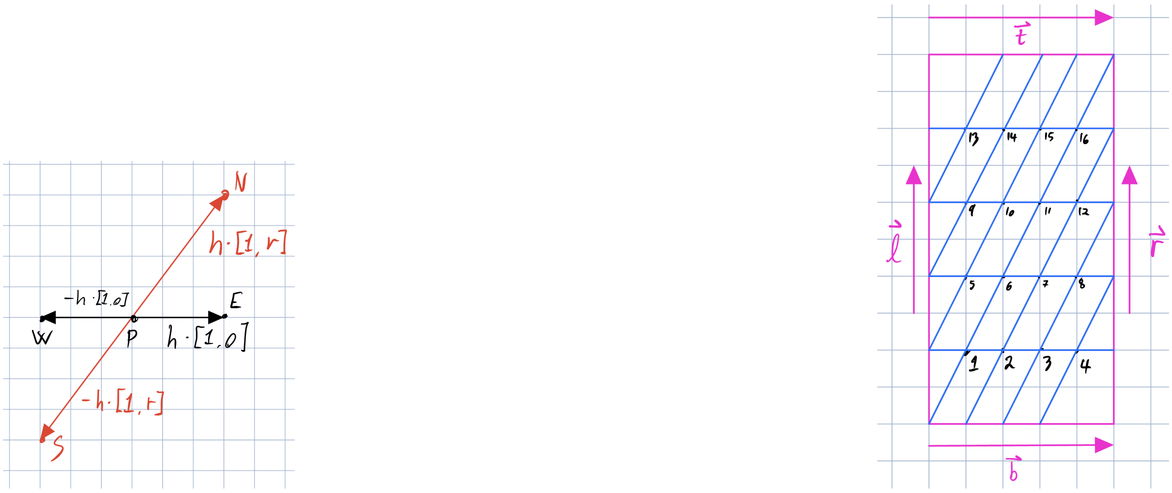
With heat conductivity κ. We are interested in the domain $\Omega = [0,1] \times [0,2]$, and we want to discretize in directions

$$\vec{d_1} = (1,0), \quad \vec{d_2} = (1,r) \tag{1.2}$$

such that we can describe heat flow in these two directions. We arrive at the equation with second-order directional derivatives in the $\vec{d_1}, \vec{d_2}$ directions:

$$a\partial_x^2 u + (\vec{d_2} \cdot \nabla^2)u = -f \tag{1.3}$$

Here, $a/(1+r^2)$ represents the relative strength of conductivity in the $\vec{d_1}, \vec{d_2}$ directions. Let's discretize this using second-order central differences these directions. Firstly, we study the stencil and grid:



(a) Sketch of stencil.

(b) Sketch of grid with $r = 2$.

And arrive at

$$\begin{aligned} a\frac{1}{h^2}\delta_x^2 U_P + \frac{1}{h^2}\delta_{d_2}^2 U_P &= -f_P \quad \text{for } P \in \mathbb{G} \\ a(U_E - 2U_P + U_W) + (U_N - 2U_P + U_S) &= -h^2 f_P \\ -\mathcal{L}U_P := -aU_E - aU_W + 2(a+1)U_P - U_N - U_S &= h^2 f_P \end{aligned} \tag{1.4}$$

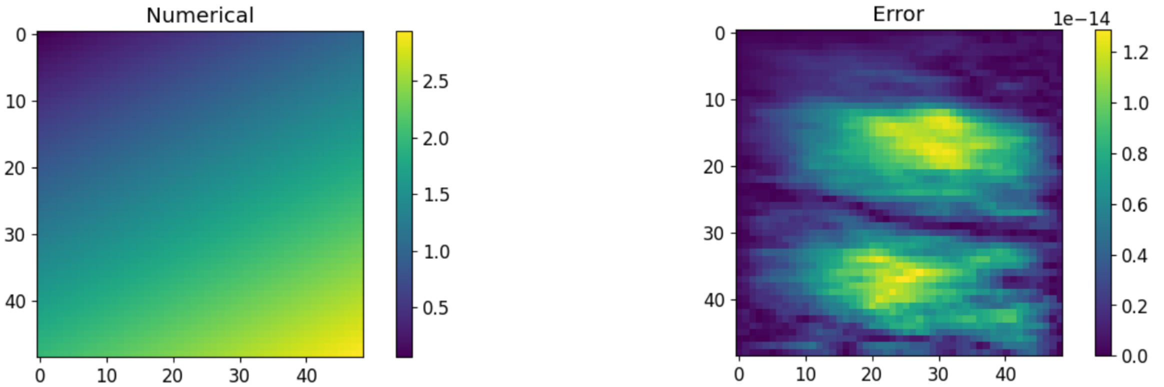
We want to write this in the form of a linear equation, $A\vec{U} + \vec{g} = -h^2\vec{f}$ (where $\vec{g}$ is the contribution from boundary conditions). We need to enumerate the nodes, so we use natural ordering as described in the sketch above. We see from the sketches that A should consist of block matrices

$$A = \begin{bmatrix} C & B & & \\ D & C & \ddots & \\ & \ddots & \ddots & B \\ & & D & C \end{bmatrix} \tag{1.5}$$

Where $C, B, D \in \mathbb{R}^{(M-1)\times(M-1)}$. And $C = \text{tridiag}(-a, 2(a+1), -a)$, $\quad B = \text{tridiag}(0, 0, -1)$, $\quad D = \text{tridiag}(-1, 0, 0)$. Let's take a look at boundary conditions $u = g \quad$ in $\partial\Omega$ Assuming that we get $\vec{r}, \vec{l}, \vec{b}, \vec{t} \in \mathbb{R}^{M+1}$ as inputs, our strategy is to construct a matrix $G \in \mathbb{R}^{(M-1)\times(M-1)}$ consisting of contributions from the boundary at each point. After this, we flatten to an $\vec{g} \in \mathbb{R}^{(M-1)^2}$. From our sketch, we can see that

- For the leftmost points, point number i gets the contribution $-al_i - l_{i-1}$.
- For the rightmost points, point number i gets the contribution $-ar_i - r_{i+1}$.
- For the bottom points, point number i gets the contribution $-b_{i-1}$.
- For the top points, point number i gets the contribution $-t_{i+1}$.

Where we will not include b_0 or t_{M+1}, as the contribution from these points are already counted ($b_0 = l_0$, $t_{M+1} = r_{M+1}$). Now, implementing this scheme, and solving for some simple boundary functions (derived from the analytic solution $u(x, y) = x + y$), we end up with the plots below.



(a) The figure shows the numerical solution of our scheme with some basic boundary functions.

(b) The figure shows the error-plot. We can see that the error is of the order of machine-error

2 Error Analysis

As of now, the numerical solver we have implemented is working. We are interested in the convergence regarding the complexity of the method. We need to evaluate the truncation error given the following problem, and we have to evaluate the directional derivatives given from equation (1.2) and Taylor expand each term in the discretisation given from equation: (1.4).

$$\begin{aligned}
&\frac{1}{h^2}(U^n_{d_2+1} - 2U^n_{d_2} + U^n_{d_2-1}) + \frac{a}{h^2} + (U^{n+1}_{d_2} - 2U^n_{d_2} + U^{n-1}_{d_2}) \\
&\frac{1}{h^2}(U^n_{d_2-1} + aU^{n-1}_{d}) - \frac{1+a}{h^2}(2U^n_{d_2}) + \frac{1}{h^2}(U^n_{d_2+1} + aU^{n+1}_{d}) \\
&\frac{1}{h^2}U_S + \frac{a}{h^2}U_W - \frac{2(1+a)}{h^2}(U_P) + \frac{1}{h^2}U_N + \frac{a}{h^2}U_E
\end{aligned} \tag{2.1}$$

and we set the coefficients as follows

$$\alpha = \frac{a}{h^2}, \quad \beta = \frac{1}{h^2} \tag{2.2}$$

We insert this into our equation, and we get the following representation of the discretisation given below. The indexes are labeled as s for south, n for north, e for east, and w for west.

$$\beta U_s + \alpha U_w - (\alpha + \beta)U_s + \beta U_n + \alpha U_e \tag{2.3}$$

We can show that the scheme given in equation (2.1) is monotone by proving that the sparsity of the matrix is diagonally dominant.

$$\mathbb{G}: \quad \beta + \alpha - 2(\beta + \alpha) + \beta + \alpha = 0, \qquad \partial\mathbb{G}: \quad \beta + \alpha - 2\beta - 2\alpha = -\beta - \alpha \tag{2.4}$$

Thus we have shown that the sparse matrix is indeed monotone, which implies the DMP (Discrete Maximum Principle) is valid for our scheme. The next thing to show is stability in L^∞. We can now use the DMP and the following comparison function given in equation (2.5). From the DMP, which states that there exists a maximum for some d_2 and n at the boundary, which are $\{U^{d_2}_n, U^{d_2}_0, U^0_{d_2}, U^m_N, 0\}$. Since we are dealing with Dirichlet boundaries, we can set them to any arbitrary constant. We assume they are zero and can perform a level shift.

$$\Phi(x) = \frac{1}{2}x(x-1) \tag{2.5}$$

This is a nice function since it is upper-bounded in the interval $[0, 1]$, and on the boundary, it is zero. We will use the following function above to prove stability in the infinity norm. Now, we can assume V solves the problem and -$\mathcal{L} = -\frac{a}{h^2}\delta_x^2 - \frac{1}{h^2}\delta_{d_2}^2 = f$. Now generate a new function W, which is defined as

$$\begin{aligned}
W &= V - \Phi \\
\mathcal{L}W &= \mathcal{L}V - \mathcal{L}\Phi
\end{aligned} \tag{2.6}$$

We isolated the function above with respect to V and we can show L^∞-stability as follows.

$$||\mathcal{L}V||_\infty = ||\mathcal{L}W + \mathcal{L}\Phi||_\infty \leq ||\mathcal{L}W||_\infty + ||\mathcal{L}\Phi||_\infty = C \tag{2.7}$$

Now we discretise Φ to find an upper bound to it. This will ensure that the function is upper bounded and properly defined; otherwise, this comparison function would be bad since we can't show that our scheme is stable in the infinity norm.

$$\mathcal{L}\Phi \approx \beta U_w + \alpha U_w - 2(\alpha + \beta)U_p + \beta U_e + \alpha U_e \leq 1 \tag{2.8}$$

Since V and Θ satisfy the DMP and the Dirichlet boundary conditions, we have the following:

$$\max||V|| \leq ||f||_\infty \tag{2.9}$$

We can find the complexity of the problem by Taylor expanding the two-dimensional problem. Thus, we get that we need to Taylor expand the following terms. Notice we have to use the multivariable definition to do this.

$$\begin{aligned} f(x + h \cdot d_i) &= f(x) + h \cdot d_i \nabla f(x) + \nabla \frac{h^2}{2} d_1 H_f d_1^T + \mathrm{O}(h^4) \\ f(x - h \cdot d_i) &= f(x) - h \cdot d_i \nabla f(x) + \nabla \frac{h^2}{2} d_i H_f d_i^T + \mathrm{O}(h^4) \end{aligned} \tag{2.10}$$

for $i = \{1, 2\}$, thus the truncation error is trivially

$$||\tau||_\infty = C \cdot h^2 ||(F^{(4)}(x)||_\infty + ||F^{(4)}(y))||_\infty \tag{2.11}$$

The F is a large multiplication product of derivatives. This is then the rate of convergence for smooth solutions. To evaluate the error, we need to find a relation between the linear operator on the error and the truncation error.

$$\begin{aligned} \mathcal{L}_h e_p &= \mathcal{L}_h u_p - \mathcal{L}_h U_p \\ &= \mathcal{L}_h u_p - f_p \\ &= f_p - \tau_h - f_p \\ &= \tau_p \end{aligned} \tag{2.12}$$

Now we can use the max norm to find an upper bound for the error. Since we have shown stability, and the scheme is von Neumann stable, the following holds.

$$||e||_\infty \leq C \cdot ||\tau||_\infty = C \cdot h^2 ||F^{(4)}(x)||_\infty + ||F^{(4)}(y)||_\infty \tag{2.13}$$

2.1 Numerical computation of error convergence

To check the convergence of the method, we will use the analytic solution

$$u(x, y) = \sin(\pi x)\cos(\pi y) \tag{2.14}$$

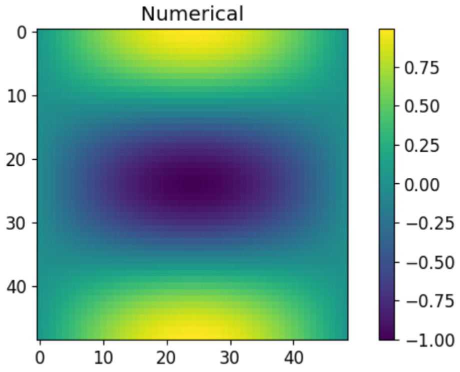
And compute f by inserting u into the equation

$$(a + 1)u_{xx} + 2ru_{xy} + r^2 u_{yy} = -f.$$

We then end up with

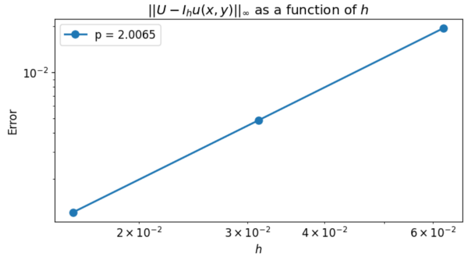
$$\begin{aligned} -f &= (a + 1)\partial_x[\pi \cos(\pi x) \cos(\pi y)] + 2r\partial_x[-\pi \sin(\pi x) \sin(\pi y)] + r^2\partial_y[-\pi \sin(\pi x) \sin(\pi y)] \\ &= (a + 1)(-\pi^2) \sin(\pi x) \cos(\pi y) + 2r(-\pi^2) \cos(\pi x) \sin(\pi y) + r^2(-\pi^2) \sin(\pi x) \cos(\pi y) \\ f &= \pi^2 \Big[(a + 1 + r^2) \sin(\pi x) \cos(\pi y) + 2r \cos(\pi x) \sin(\pi y)\Big] \end{aligned}$$

The boundary conditions can easily be retrieved from the exact solution. Now inserting this into our code, we end up with plot (a) below. By solving with the method for different values of the step length h, we can check convergence (shown in plot (b)) below.



(a) The figure shows the numerical solution of our scheme. Boundary conditions are derived from (2.14)

.



(b) The figure shows the error plot. We can see that the error is of order two.

From these plots, we can see that they nicely coincide with the theory we derived.

3 Introduction to fattening the boundary

To explore our method further, we consider a situation where r is irrational. This poses a problem since we cannot choose an h such that we hit the upper boundary, $y = 2$. There are multiple ways of solving this issue, but we went for what seemed most straightforward: fattening the boundary. This is a method to impose boundary conditions on an irregular grid. We extend the boundary so that there is one row outside of the domain (with the intention of evaluating these points as well), and then we construct a new boundary on this row. Theoretically, this should give us convergence of order one near the boundary. Let us check if this holds with $r = \pi$. We will reuse the example of (2.14). We then compare the second to last row of the numerical result with the exact solution.

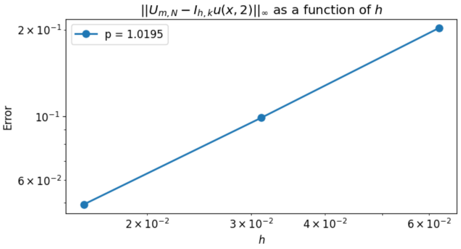


Figure 3.1: The figure shows the error convergence at the fattened boundary is one.

From these plots, we can see that the error nicely coincides with our expectations.

4 Isotropic case in an irregular domain

We now want to deal with the more complicated domain. Let Ω be the domain in the first quadrant enclosed by the parabola $y = 1 - x^2$ and the two axes. The boundary then consists of the curves $\gamma_1 = [0,1] \times \{0\}$, $\gamma_2 = \{0\} \times [0,1]$, and $\gamma_3 = \{(x, 1-x^2) : x \in [0,1]\}$. We want to numerically solve the Dirichlet boundary value problem for our model in the isotropic case when $\kappa = I$ and

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

There are two ways of handling irregular boundaries (that we want to test). We can either modify the discretisation near the boundary or fatten the boundary. We will first look at the discretisation modification. In the places where branches of the stencil intersect with the hyperbola, we need to modify the discretisation. Assuming that the grid point, P, in the middle of the stencil is on the inside of our domain, only the north and east points of the stencil will, at any point, be outside of the hyperbolic boundary. We will therefore define our modified stencil points as $E' = P + (\xi h, 0)$, and $N' = P + (0, \eta h)$, where $\xi, \eta \in (0,1)$. To create our modified discretisation, we will use the method of undetermined coefficients, see appendix A. Solving this system, we end up with the coefficients

$$a_{E'} = \frac{2}{\xi h^2(1+\xi)}\ , \quad a_W = \frac{2}{h^2(1+\xi)}\ , \quad a_P = -\frac{2}{\xi h^2}. \tag{4.1}$$

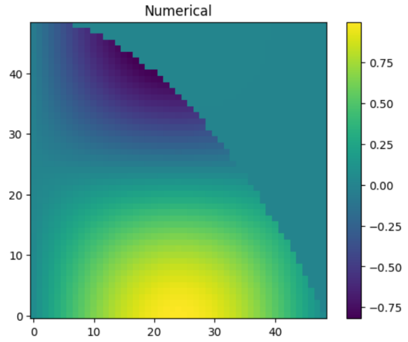
Doing the exact same method in the y-direction, we end up with the coefficients

$$a_{N'} = \frac{2}{\eta h^2(1+\xi)}\ , \quad a_S = \frac{2}{h^2(1+\eta)}\ , \quad b_P = -\frac{2}{\eta h^2}. \tag{4.2}$$

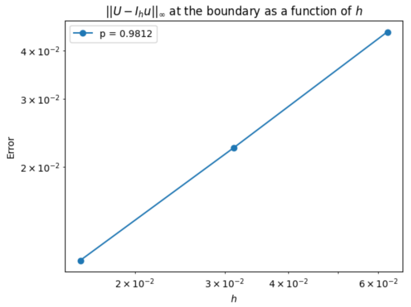
For simplicity in the code, we want the h^2-term to be on the right side of the equation below, so we redefine our coefficients not to have it. Our modified discretisation becomes:

$$-\mathcal{L}_h U_P := -a_W U_W - a_{E'} U_{E'} - a_S U_S - a_{N'} U_{N'} + (a_P + b_P) U_P = h^2 f_P \tag{4.3}$$

Implementing this scheme, using the same analytical equations as above,(2.14), our result becomes:



(a) The figure shows the numerical solution of our modified stencil scheme. The points outside of the domain have been manually set to zero.



(b) The figure shows the error convergence at the boundary of our modified stencil scheme.

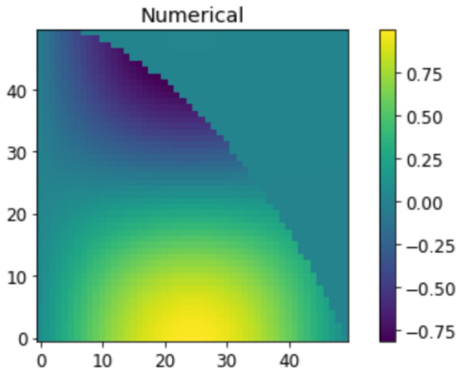
Now, we will evaluate the second method; fattening the boundary. We need to evaluate which points fall outside our boundary (x_p, y_p) and evaluate the projection of those points down on the boundary. These new projection points can be evaluated by solving for r from the following equation

$$x_p + r \cdot (1 - 2 \cdot y_p) - 2r^3 = 0, \tag{4.4}$$

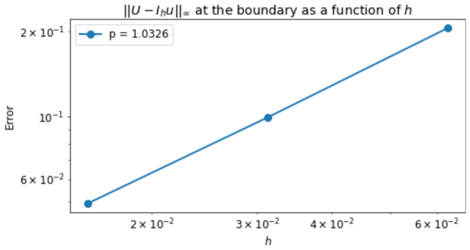
where r is some real positive solution. When we have solved for r, we can then evaluate the new point on the boundary by using the following equation below.

$$X_P = (r, 1 - r^2) \tag{4.5}$$

Then we can evaluate the new points that are exactly at the boundary and use them as the points that fall outside the boundary. These new points are evaluated using equation (4.5) when we solve equation (4.4) for some r where x_p and x_y is the actual point outside the boundary. These new evaluated points of the function are then put inside the g-vector, which contains every boundary point evaluated.



(a) The figure shows the numerical solution of our boundary-fattening scheme. The points outside of the domain have been manually set to zero.



(b) The figure shows the error convergence at the boundary of our boundary-fattening scheme.

Now, which of the two methods is faster to implement? In our case, in 2 dimensions, the two methods were more or less comparable in complexity. The problem with the stencil modification is that the more points the stencil include, for example, in higher dimensions, the amount of different modified stencils grows factorially. It would be much easier to implement boundary-fattening because each point of the stencil that "sticks out" of the boundary requires only a projection and a function evaluation, growing exponentially in complexity.

5 Appendix A

We will begin by looking at the x-direction. Solving this system, we end up with the coefficients

$$\begin{aligned}\partial_x^2 u_P - \tau_{xP} &= a_W u_W + a_P u_P + a_{E'} u_{E'} \\ &= a_W u(x_P - h, y_P) + a_P u(x_P, y_P) + a_{E'} u(x_P + \xi h, y_P)\end{aligned}$$

Taylor-expanding this equation, we get

$$\begin{aligned}\partial_x^2 u_P - \tau_{xP} = a_W &(u - h u_x + \frac{1}{2} h^2 u_{xx} + \mathcal{O}(h^3) \\ &+ a_P u \\ &+ a_{E'} (u + \xi h u_x + \frac{1}{2} \xi^2 h^2 u_{xx} + \mathcal{O}(\xi^3 h^3)\end{aligned}$$

This gives us a system of equations to solve

$$\begin{aligned}a_W + a_P + a_{E'} &= 0 \\ h(\xi a_{E'} - a_W) &= 0 \\ \frac{h^2}{2}(\xi^2 a_{E'} + a_W) &= 0 \\ a_W \mathcal{O}(h^3) + a_{E'} \mathcal{O}(\xi^3 h^3) &= \tau_{xP}\end{aligned}$$