NTNU
Kunnskap for en bedre verden

Modelling of stationary 2D temperature distribution in anisotropic and isotropic materials

TMA4212

NUMERICAL SOLUTION OF DIFFERENTIAL EQUATIONS BY DIFFERENCE METHODS

Authors:
Heidi Bjørnerud Vatnøy, Elias Rekkedal, Max Pfisterer

26.02.23

Table of Contents

1 Summary **1**

2 Abstract **1**

3 Heat distrubution in anisotropic materials **2**

3.1 Discretisation and trunctation error 2

3.2 Monotonicity 3

3.3 L^∞-stability 4

3.4 Error analysis 4

3.5 Testing the scheme, convergence rate and error bound 5

4 Heat distrubution in isotropic materials **5**

4.1 Discretisation 5

4.2 Numerical results 6

5 Who did what? **6**

Appendix **7**

A Anisotropic Materials 7

B Isotropic materials 10

B.1 Fattening the boundary 10

B.2 Modified scheme 11

1 Summary

In this project we have looked at numerical solutions of the Poisson equation modelling stationary temperature distribution T for a solid domain Ω with Dirichlet boundary conditions

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

In the case of anisotropic materials we looked at heat flow of different rates in two directions on a square grid, $\vec{d_1} = (1,1)^T$ and $\vec{d_2} = (1,r)^T$. We used central differences to approximate the the equation, and solved it as a linear system. Both the analytical and numerical results gave a convergence rate of second order. We then used a modified scheme to numerically model the same situation where one of the flow directions was irrational, i.e. the r in $\vec{d_2}$ was irrational. The modified scheme we used in this case also had a quadratic rate of convergence. In the anisotropic case, we modeled the flow to be ten times stronger in the direction of $\vec{d_2}$.

When looking at isotropic materials, $\kappa = I$, we solved the equation on a non-regular grid. To do this we used two methods; a modified scheme and fattening the boundary. When solving with the modified scheme, similar to the anisotropic case, we found a second order numerical rate of convergence, whereas when we used fattening the boundary it was of first order.

2 Abstract

For a solid domain Ω, we can model the stationary temperature distribution T with the Poisson equation, given a heat conductivity κ, and internal heat sources f. The following model is derived from the conservation of energy, Fourier's law for heat flux and the stationary property ($\partial_t T = 0$):

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

In materials that do not have a uniform heat flow, i.e. that are anisotropic, we get that κ is a matrix. We will model a two-dimensional case where the heat flows in the directions: $\vec{d_1} = (1,0)^T$ and $\vec{d_2} = (1,r)^T$, $r \in \mathbb{R}$. Normalising, we get the heat conductivity matrix:

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

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 $\vec{d_1}$ compared to the $\vec{d_2}$ direction. In this case problem 1 becomes:

$$\begin{aligned} &-\nabla \cdot (\kappa \nabla T) = f \Leftrightarrow \\ &-\big((a+1)\partial_x^2 u + 2r\partial_x \partial_y u\big) = f \Leftrightarrow \\ &-\left(a\partial_x^2 u + \left(\vec{d_2} \cdot \nabla\right)^2 u\right) = f \end{aligned} \tag{3}$$

In the first part of the project we will consider equation 3 on a domain $\Omega = [0,1] \times [0,2]$ with Dirichlet boundary conditions $u = g$ on $\partial\Omega$. The problem will be solved numerically using finite difference methods, and we will do a convergence analysis of the scheme both analytically and numerically.

We will also study some different ways of handling irregular domains Ω, meaning domains with non-straight borders. Approximating derivatives along the border will then be more complex when implementing finite difference methods. We will implement and test methods for solving the simpler, isotropic stationary heat equation

$$-\Delta T = -\nabla \cdot \nabla T = f \tag{4}$$

for the non-regular domain on the positive quadrant enclosed by $y = 1 - x^2$. In one method we will create new grid points at the intersection of the grid lattice and the border and derive an alternative scheme, and derive an alternative scheme along the border. In another, we will extend our domain to also include the grid-points just outside the original border, called ”fattening the boundary”. We will analyze convergence rate for the two empirically.

3 Heat distrubution in anisotropic materials

3.1 Discretisation and trunctation error

To solve equation 3 on $\Omega = [0,1] \times [0,2]$ with Dirichlet boundary conditions $u = g$ on $\partial\Omega$ we use a finite difference method. This entails defining a grid over our domain, approximating the derivatives on each inner grid point with a finite difference (i.e. with the other grid points), and thus solve a linear system instead of a PDE.

First, we consider the case where $\vec{d_2} = (1,2)$. We can then utilize the fact that the length of the domain in the y direction is divisible by the y-component of $\vec{d_2}$ and define a regular grid with step sizes $h = \frac{1}{M}$ $, M \in \mathbb{N}$, and $k = 2h$ in the x and y directions respectively. This gives us the opportunity to approximate the directional derivative with the directional central difference:

$$\left(\vec{d_2} \cdot \nabla\right)^2 u_m^n = \partial_x^2 u_m^n + 2r\partial_x\partial_y u_m^n \approx \frac{1}{h^2}(au_{m+1}^{n+1} + bu_m^n + cu_{m-1}^{n-1}) =: \frac{1}{h^2}\delta_{\vec{d_2}}^2 u_m^n \tag{5}$$

Where $u_m^n = u(m,n)$. By doing a Taylor expansion (for general r) of u_{m+1}^{n+1} and u_{m-1}^{n-1} around (x,y) we get the coefficients a, b and c, as well as the truncation error $(\tau_{\vec{d_2}})_m^n$:

$$\begin{aligned}
&(\tau_{\vec{d_2}})_m^n = \left(\vec{d_2} \cdot \nabla\right)^2 u_m^n - \frac{1}{h^2}\delta_{\vec{d_2}}^2 u_m^n = \partial_x^2 u + 2r\partial_x\partial_y u \\
&- \frac{1}{h^2}(a(u_m^n + (h\partial_x + rh\partial_y)u_m^n + \frac{1}{2}(h\partial_x + rh\partial_y)^2 u_m^n + \frac{1}{3!}(h\partial_x + rh\partial_y)^3 u_m^n + \frac{1}{4!}(h\partial_x + rh\partial_y)^4 u_m^n) \\
&+ c(u_m^n + (-h\partial_x - rh\partial_y)u_m^n + \frac{1}{2}(-h\partial_x - rh\partial_y)^2 u_m^n + \frac{1}{3!}(-h\partial_x - rh\partial_y)^3 U_m^n + \frac{1}{4!}(h\partial_x + rh\partial_y)^4 u_m^n) \\
&+ bu_m^n + \mathcal{O}(h^5)) = \frac{h^2}{12}(\partial_x + r\partial_y)^4 u_m^n + \mathcal{O}(h^3)
\end{aligned}$$

With the coefficients $a = 1, b = -2, c = 1$. To approximate the x directional derivative, we approximate with the x-direction central difference:

$$a\partial_x^2 u_m^n \approx \frac{a}{h^2}\delta_x^2 u_m^n := \frac{a}{h^2}(u_{m+1}^n - 2u_m^n + u_{m-1}^n) \tag{6}$$

with known truncation error $(\tau_x)_m^n = \frac{ah^2}{12}\partial_x^4 u_m^n + \mathcal{O}(h^3)$ We then get the scheme:

$$\mathcal{L}_h U_m^n = -\frac{1}{h^2}\left(a(U_{m+1}^n - 2U_m^n + U_{m-1}^n) + (U_{m+1}^{n+1} - 2U_m^n + U_{m-1}^{n-1})\right) \approx -\left(a\partial_x^2 u + \left(\vec{d_2} \cdot \nabla\right)^2 u\right)_m^n \tag{7}$$

with the truncation error

$$\tau_m^n = (\tau_{\vec{d_2}})_m^n + (\tau_x)_m^n = \frac{h^2}{12}(a\partial_x^4 + (\partial_x + r\partial_y)^4)u_m^n \tag{8}$$

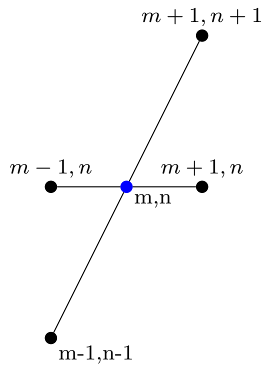


Figure 1: A picture of the stencil of our scheme.

To solve equation (3) numerically we write the problem, using the discretisation, as the matrix equation

$$A_h\vec{U} + \vec{b} = \vec{F} \tag{9}$$

where $\vec{U}$ is a vector containing some order of all the interior points on our grid. Matrix A_h contains the coefficients of the scheme corresponding to the vector $\vec{U}$. If the solution of the scheme for the interior point i is dependent upon boundary conditions, one can find these values in $\vec{b}$. $\vec{F}$ contains the right hand side values in the grid points corresponding with $\vec{U}$.

Further, we want so solve (3) for a general $\vec{d_2} = (1, r)$, $r \in \mathbb{R}$. To do this, we discretize with grid width h and $k = |r|h$ in the x- and y-direction. Then we can still use our approximation (5) for the directional derivative. However, the height of our domain Ω is no longer necessarily divisible by k. We therefore need a new way of handling the boundary on the side with a remainder. There are multiple ways of doing this. The method of fattening the boundary is explored later on. Here, we derive a new scheme for the points closest to this boundary (the upper, using standard ordering.).

Let p be the remainder of $\frac{2}{k}$, and $\eta = \frac{\sqrt{p^2+1}}{\sqrt{r^2+1}}$ be the proportional size of the last directional step. Then ${u_{m+1}}^{n+1} := u^n_m + \eta h\vec{d_2}$ will be on the boundary. Again, by Taylor expanding the proposed scheme $\delta^2_{\vec{d_2}} u^n_m = \frac{a'(u^{n+1}_{m+1})' + bu^n_m + cu^{n+1}_{m-1}}{h^2}$ we can obtain the coefficients:

$$\begin{aligned} &a'(u^n_m + \eta h(\vec{d_2}\cdot\nabla)u^n_m + \eta^2h^2(\vec{d_2}\cdot\nabla)^2u^n_m) + bu^n_m + c(u^n_m - h(\vec{d_2}\cdot\nabla)u^n_m + h^2(\vec{d_2}\cdot\nabla)^2u^n_m) \\ &\implies a = \frac{2}{\eta^2+\eta}, b = \frac{-2}{\eta}, c = \frac{2}{\eta+1} \end{aligned} \tag{10}$$

These values can then be inserted into their correct spaces in A_h and b.

3.2 Monotonicity

A scheme is monotone if it has positive coefficients. A scheme written like

$$-\mathcal{L}_hU_p = \alpha_{PP} - \sum \alpha_{PQ}U_Q = F_P,$$

has positive coefficients if it satisfies

$$\alpha_{QP} \geq 0 \text{ and } \alpha_{PP} \geq \sum \alpha_{PQ}. \tag{11}$$

From our scheme we get

$$\alpha_{PP} = \alpha^n_m = 2(a+1), \alpha^{n-1}_{m-1} = \alpha^{n+1}_{m+1} = 1 \geq 0, \alpha^n_{m-1} = \alpha^n_{m+1} = a \geq 0.$$

$$\alpha_{m-1}^{n-1} + \alpha_{m+1}^{n+1} + \alpha_{m-1}^{n} + \alpha_{m+1}^{m} = 2 + 2a = 2(1+a) = \alpha_m^n. \tag{12}$$

The conditions for positive coefficients are satisfied, thus making it monotone.

Since $\mathcal{L}_h$ has positive coefficients and is boundary connected, it is monotone.

3.3 L^∞-stability

Since $\mathcal{L}_h$ is monotone we know that the discrete maximum principle (DMP) holds:

$$\max_{\mathbb{G}} V_p \leq \max_{\partial\mathbb{G}} \{V_p, 0\}. \tag{13}$$

$$\phi(x) = \frac{x}{2}(1-x). \tag{14}$$

We will now show stability 15 for our scheme by using a $V = 0$ on $\partial\mathbb{G}$ that solves $-\mathcal{L}_h V = f_h$ and our super solution 14, which is 0 on the boundary and satisfies $-\mathcal{L}_h\phi \geq 1$ and $\max\limits_{\Omega}\phi = \frac{1}{8}$:

$$\begin{aligned}
-\mathcal{L}_h\phi &= -\frac{a+1}{h^2}\delta_x^2 = \frac{a+1}{h^2}\left(2\phi(x) - \frac{1}{2}(\phi(x+h) + \phi(x-h)\right) \\
&= \frac{1+a}{h^2}\left(x - x^2 - \frac{1}{2}((x+h) - (x+h)^2 + (x-h) - (x-h)^2)\right) \\
&= \frac{1+a}{h^2}\left(x - x^2 - \frac{1}{2}(2x - (2x^2 + 2h^2))\right) \\
&= 1 + a \geq 1
\end{aligned}$$

$$\begin{aligned}
&-\mathcal{L}_h(V_p - \|\vec{f}\|_\infty\phi_p) = f_p - \|\vec{f}\|_\infty(-\mathcal{L}_h\phi_p) \\
&= f - (1+a)\|\vec{f}\|_\infty \leq 0 \\
&\overset{13}{\Longrightarrow}\ V_p - \|\vec{f}\|_\infty\phi_p\ \leq \max_{\partial\mathbb{G}}\{V_p - \|\vec{f}\|_\infty\phi_p, 0\} = 0 \\
&\Longrightarrow\ V_p \leq \frac{1}{8}\|\vec{f}\|_\infty \overset{V,f\rightarrow -V,-f}{\Longrightarrow} \|V\|_\infty \leq \frac{1}{8}\|\vec{f}\|_\infty.
\end{aligned} \tag{15}$$

3.4 Error analysis

Having proven stability of our scheme in 7, we want to use the truncation error to then get a bound for the global error. We first start with expanding the expression for the truncation error from equation 8 and inserting $r = 2$:

$$\begin{aligned}
\tau_m^n &= \frac{h^2}{12}(a\partial_x^4 + (\partial_x + r\partial_y)^4)u_m^n \\
&= (a\partial_x + (\partial_x^4 + 4r\partial_x^3\partial_y + 6r^2\partial_x^2\partial_y^2 + 4r^3\partial_x\partial_y^3 + r^4\partial_y^4))u_m^n \\
&= (a\partial_x + (\partial_x^4 + 4r\partial_x^3\partial_y + 24\partial_x^2\partial_y^2 + 32\partial_x\partial_y^3 + 16\partial_y^4))u_m^n.
\end{aligned}$$

To get an error bound for our method we can use the stability result 15 and the fact that $e = u - U = 0$ on $\partial\mathbb{G}$:

$$\begin{aligned}
&-\mathcal{L}_h\vec{e_p} = -\mathcal{L}_\langle(u - U) = \tau_p \\
&\Longrightarrow\ \max_p\|\vec{e_p}\|_\infty \leq \frac{1}{8}\max_p\|\tau_p\|_\infty \\
&\Longrightarrow\ \max_p\|\vec{e_p}\|_\infty \leq \frac{h^2}{96}\|(1+a)u_{xxxx} + 8u_{xxxy} + 24u_{xxyy} + 32u_{xyyy} + 16u_{yyyy}\|_\infty.
\end{aligned} \tag{16}$$

This means that we for sufficiently smooth solutions, having at least one well-defined fourth order partial derivative, get a quadratic rate of convergence ($\mathcal{O}(h^2)$) for the scheme in equation 7 in 3.1.

3.5 Testing the scheme, convergence rate and error bound

We have used the following test functions:

$$u_1(x,y) = \cos(x\pi)\sin(y\frac{\pi}{2}),\;\; u_{f1}(x,y) = \pi^2\left(\cos(x\pi)\sin(y\frac{\pi}{2})((a+1)+\frac{r}{2}^2) + r\sin(x\pi)\cos(y\frac{\pi}{2})\right),$$
$$u_2(x,y) = y\sin(x\pi),\;\; u_{f2}(x,y) = \pi^2 y\sin(x\pi)(a+1) - 2\pi r\cos x\pi. \tag{17}$$

We have plotted the numerical solutions, the error, and convergence rates for both cases, using the suitable right hand sides $\vec{F}$. These plots are in Appendix A.

We have experimentally confirmed second order convergence of our scheme, as predicted from the taylor expansion and error analysis in 16. We have also plotted our error bound, where we found the fourth order derivatives and evaluated the normed term in equation 16. We see that for u_1 and u_2, in the figures referenced earlier. The maximal error estimate is quite conservative, as we bound the global error by the largest truncation error, but we only get an error this large at a few points close to the boundary, and it is much smaller otherwise. The function that we bound the truncation error by is largest on the x=0 boundary, and then quickly decreases, as we get sinusoid terms of order π^4.

4 Heat distrubution in isotropic materials

4.1 Discretisation

Now we will solve the stationary heat equation on a uniform grid with $h = \frac{1}{M}$, and $k = h$ the step sizes in the x and y direction respectively. We now let the domain Ω be the set of points in the area enclosed by the two axes and the parabolic curve $y = 1 - x^2$. We can divide the boundary into:

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

To solve problem 1 in the isotropic case, where $\kappa = I$, we get:

$$\begin{aligned} &-\nabla\cdot(\kappa\nabla T) = f \Leftrightarrow \\ &-\Delta T = f \Leftrightarrow \\ &-u_{xx} - u_{yy} = f. \end{aligned} \tag{18}$$

We can then approximate as follows:

$$-\left(u_{xx} - u_{yy}\right)_m^n \approx -\frac{1}{h^2}\left((u_{m+1}^n - 2u_m^n + u_{m-1}^n) + (u_m^{n+1} - 2u_m^n + u_m^{n-1})\right). \tag{19}$$

As we here have an irregular domain, discretised with a regular grid, we will again run into the problem of the stencil reaching out of our domain for the gridpoints closest to the border. There are again several ways of working around this.

As we did on the domain $[0,1]\times[0,2]$, we can modify the scheme to use the points where the boundary $\partial\Omega$ intersects the grid lattice. For example, if u_{m+1}^n is without our boundary, one will use the value $(u_{m+1}^n)' := u(x_m + \eta h, y_n)$, where $\eta \in (0,1)$ is such that $(x_m + \eta h, y_n)$ is on the boundary, or conversely for u_m^{n+1} in the y-direction. Our approximation will then be on the form

$$\partial_x^2 u_m^n \approx \frac{a'(u_{m+1}^n)' + bu_m^n + cu_{m-1}^n}{h^2}$$

This is a very similar form as in (10). In fact, in Taylor expanding, one will get the exact same calculation, and again find the coefficients $a = \frac{2}{\eta+\eta^2}, b = -\frac{2}{\eta}$ and $c = \frac{2}{1+\eta}$. For implementing this, we must check for each point in our grid whether it has any neighbours outside of Ω, and if yes, update the corresponding values in A_h and b.

Another way of solving the issue is to "fatten the boundary". Here, when the scheme calls for the use of a point outside of Ω, we make a projection from this point onto the boundary. The value on the boundary of the projected point is the value that will be used in the vector $\vec{b}$. Otherwise the scheme stays the same.

4.2 Numerical results

For the modified scheme, we get quadratic convergence. It is worth noting that we only have linear convergence at the boundary, which follows from the loss of symmetry of our scheme for points adjacent to the boundary. It seems that since we have quadratic convergence everywhere else, that this does not affect the order of convergence for our example functions. It would be interesting to examine whether this holds for all sufficiently smooth functions or only for our functions, which have a small truncation order close to γ_3. As expected, we in all cases get that the largest error is away from the boundary, close to the center of our domain. Intuitively, this is because we control the error with boundary conditions at the boundary.

In the alternative method, using a fattening of the boundary, we get linear convergence. In theory, and as discussed in class, this is expected. Initially we were surprised that the largest errors were at the points adjacent to the boundary. We believe this is due to the nature of the method; just using the closest boundary point we can find when we have no gridpoints on the boundary. This does of course lead to error spikes at points adjacent to the boundary. These errors decrease linearly, and are large enough to prevent us from attaining quadratic convergence. As an upside, it is much easier to implement fattening the boundary, than to do calculations by hand to create a new scheme near the boundary.

We find that modifying the scheme for our domain and types of functions is more efficient with respect to computation time, as we have quadratic convergence, and therefore find a more accurate solution faster. A drawback of this method however, is that it requires calculations by hand and more effort in programming. We conclude that which method is preferred needs to be evaluated on a case by case basis. If a numerically accurate solution is important, or computation time is limited, we would prefer modifying the scheme. If we, on the other hand, wanted a more qualitative view of the solution, or wanted to experiment with different domains, or simply want a more flexible code, we would try to fatten the boundary first. It is also important to mention that a modified scheme may be much more difficult to design and implement if we have a more complex domain than in this project, so fattening the boundary may be easier in those cases.

The plots of the numerical solution, along with a corresponding plot of the error and a plot of the convergence rate of the error using a fattening of the boundary is in appendix B.1. Corresponding plots for the modified solution is in appendix B.2. All plots use $u_1(x, y)$ as the exact solution.

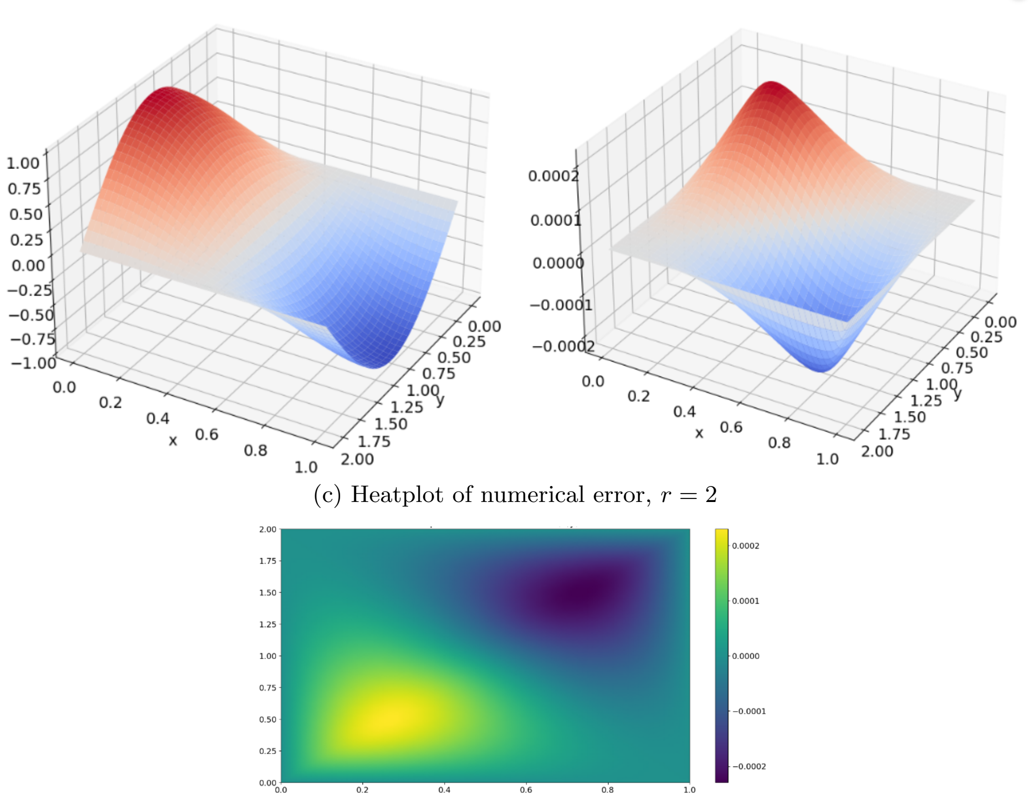
5 Who did what?

1. Heidi: Discretisation, error analysis, implementation of fattening the boundary
2. Elias: Implementation of standard and modified scheme for the anisotropic case. Modified scheme for the isotropic case.
3. Max: Stability and global error analysis, testing, convergence rates. Assisted with implementations.

Appendix

A Anisotropic Materials

Figure 2: Numerical solution and error plots for u1(x,y), $r = 2$



(a) Numerical solution

(b) Numerical error for u1(x,y), $r = 2$

(c) Heatplot of numerical error, $r = 2$

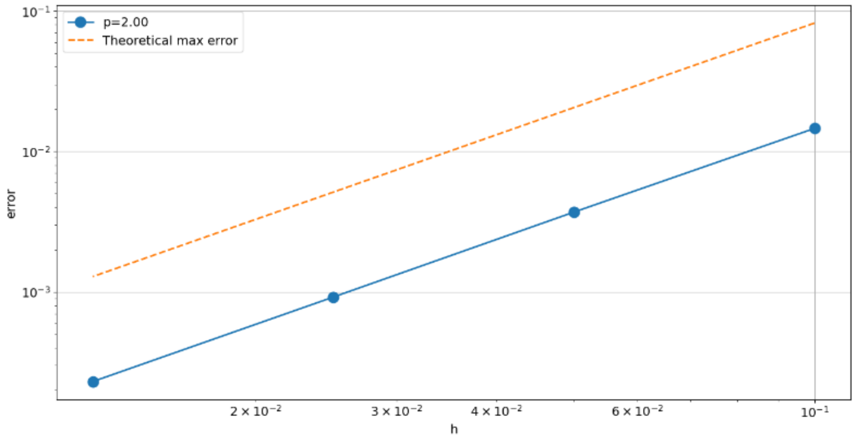
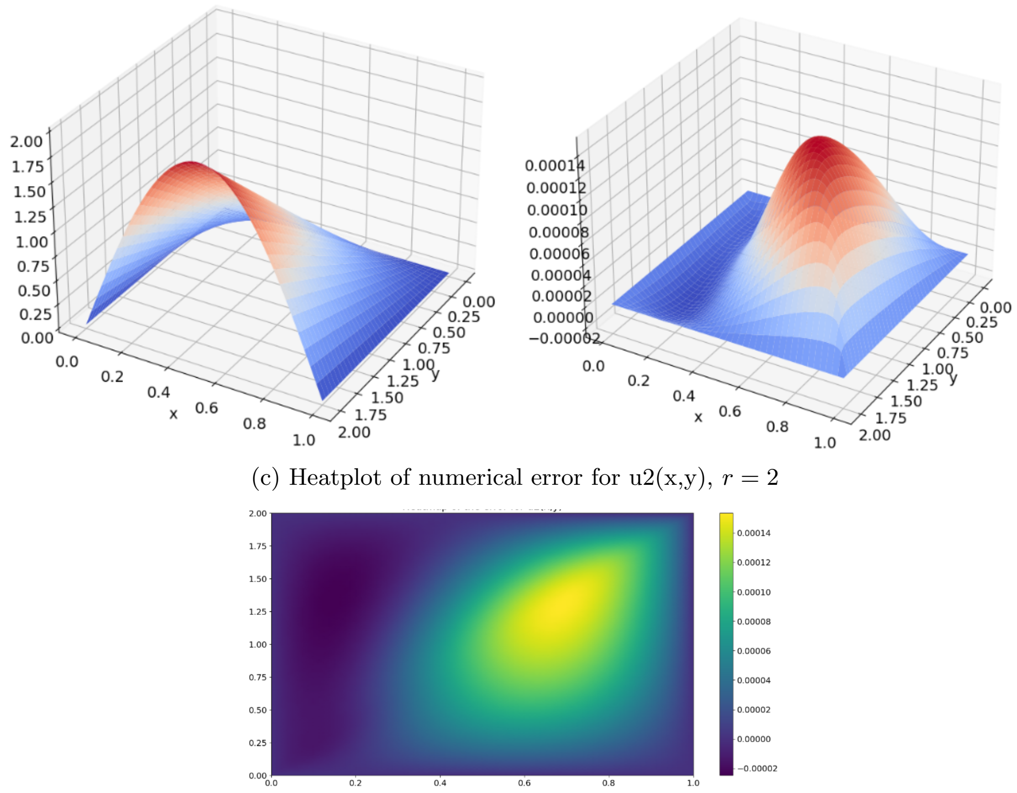


Figure 3: Rate of convergence for u1(x,y), $r = 2$

Figure 4: Numerical solution and error plots for u2(x,y), $r = 2$



(a) Numerical solution

(b) Numerical error for u2(x,y), $r = 2$

(c) Heatplot of numerical error for u2(x,y), $r = 2$

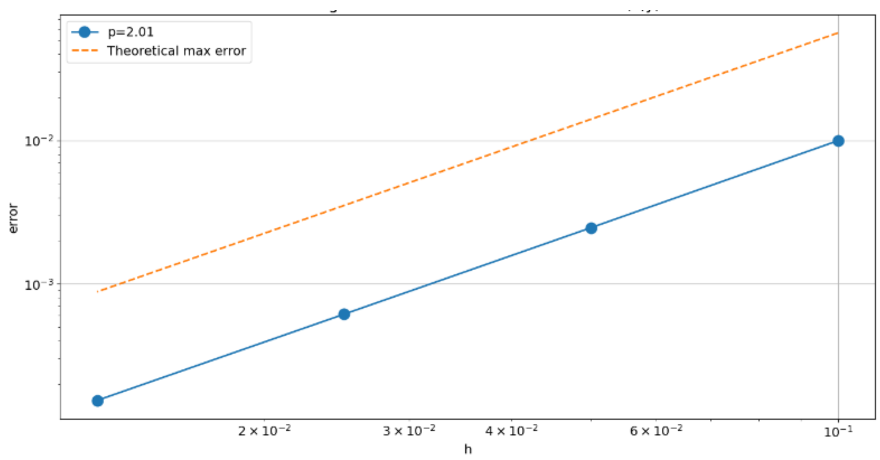
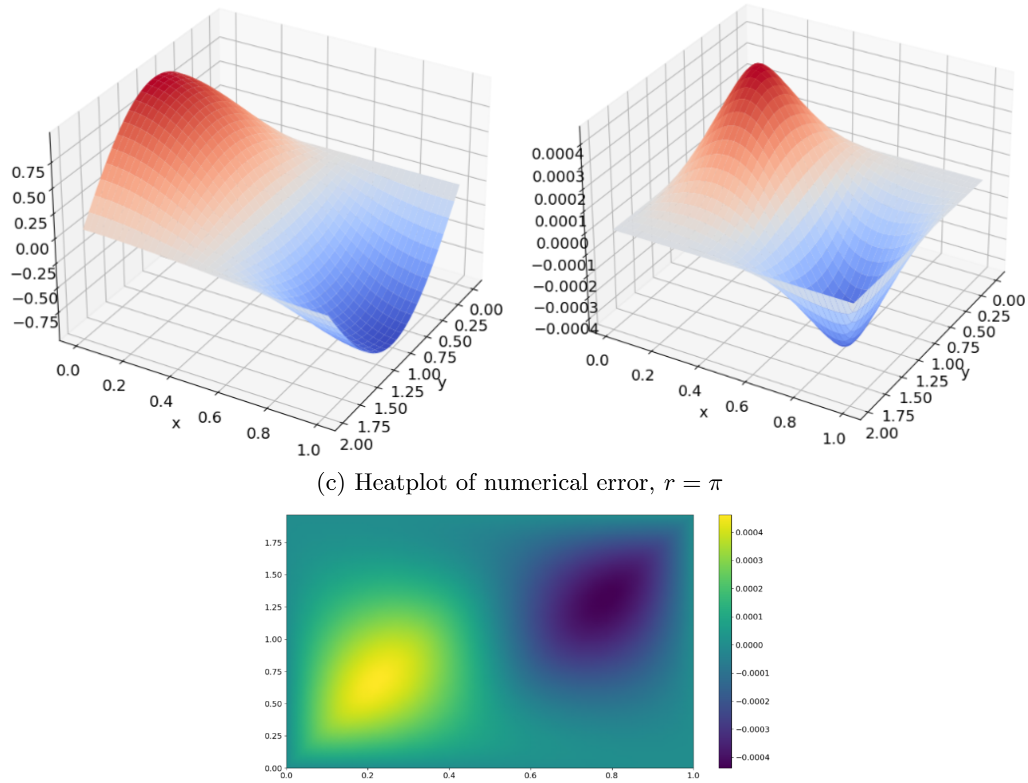


Figure 5: Rate of convergence for u2(x,y), $r = 2$

Figure 6: Numerical solution and error plots for u2(x,y), $r = \pi$

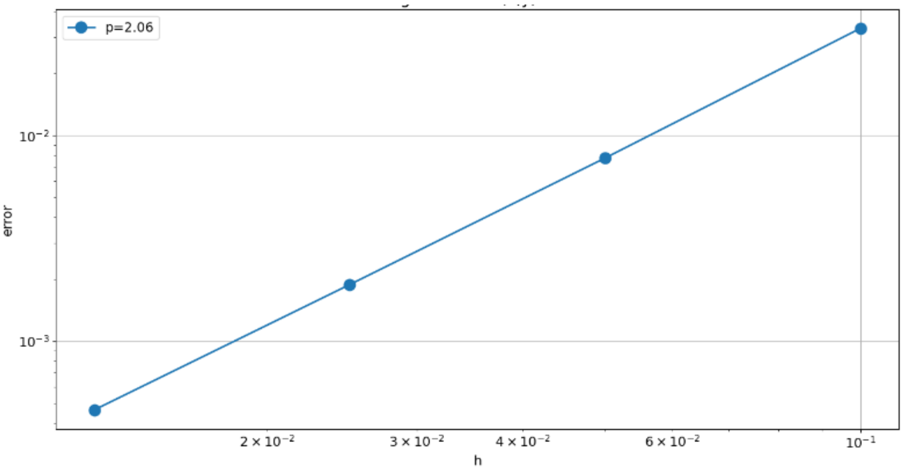


(a) Numerical solution, $r = \pi$

(b) Numerical error, $r = \pi$

(c) Heatplot of numerical error, $r = \pi$

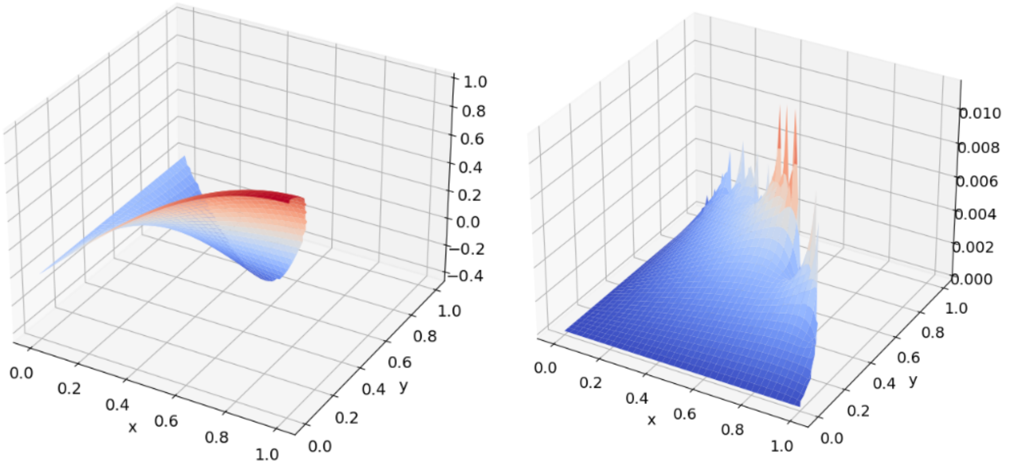
Figure 7: Rate of convergence, $r = \pi$



B Isotropic materials

B.1 Fattening the boundary

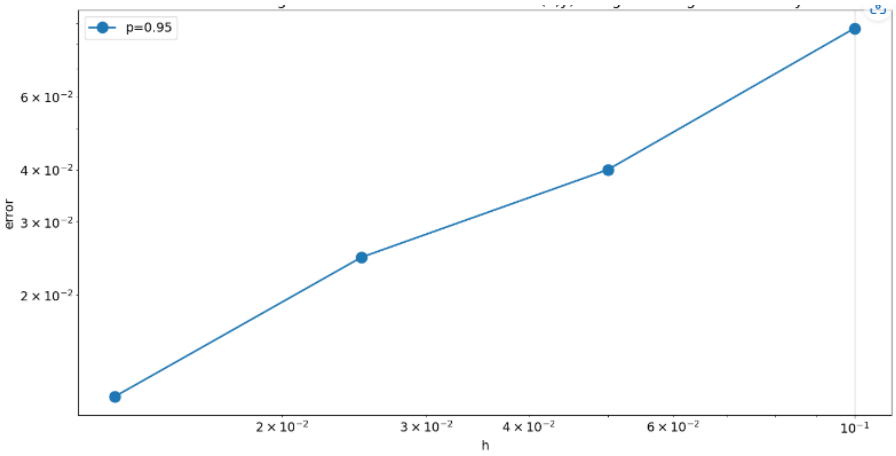
Figure 8: Numerical solution and error plots for u1(x,y)



(a) Numerical solution

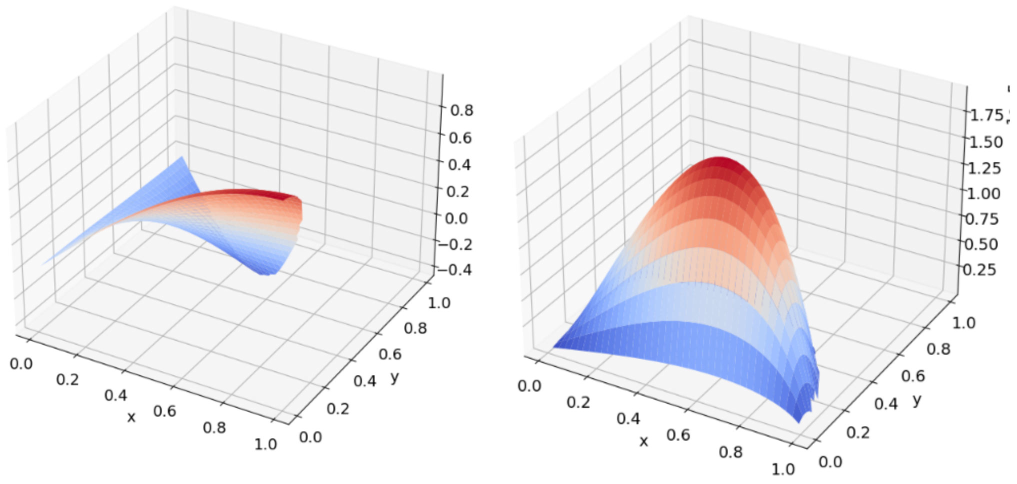
(b) Numerical error

Figure 9: Rate of convergence for u1(x,y)



B.2 Modified scheme

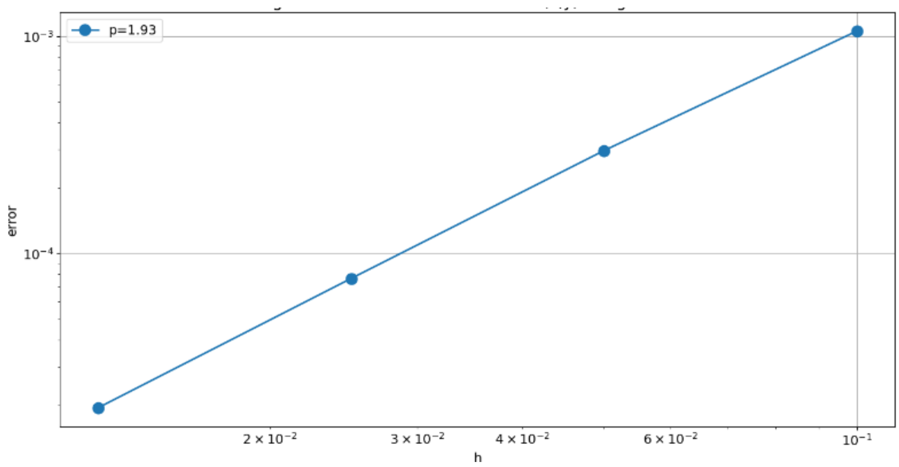
Figure 10: Numerical solution and error plots for u1(x,y)



(a) Numerical solution

(b) Numerical error

Figure 11: Rate of convergence for u1(x,y)



p=1.93
10^{-3}
10^{-4}
error
2×10^{-2}
3×10^{-2}
4×10^{-2}
6×10^{-2}
10^{-1}
h