TMA4212 - Numerical solution of differential equations by difference methods: Project 1

Aksel Stenvold, Casper Lindeman, Gonchigsuren Bor

February 25, 2024

Heat distribution in anisotropic and isotropic materials

The heat equation is an important mathematical model that is used for many different scientific approaches to solving problems in various fields. In this report, we solve the heat equation problem in anisotropic materials by developing using finite central differences. We first start by developing a model that solves the heat equation on a square boundary. Thereafter, we develop a scheme that can handle irregular boundaries.

We start by taking a closer look at the Poisson equation. We model the stationary temperature distribution T of a solid Ω with heat conductivity κ and internal heat sources f. With Fourier's law for the heat flux and stationarity, we solve for the following model

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

where

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

where $r \in \mathbb{R}$. Define $\vec{d_1} = (1, 0)$ and $\vec{d_2} = (1, r)$. The derivative part of the problem (1) can then be written as

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

Derivation of the scheme for the heat distribution problem

We will now proceed by deriving a scheme that solves the heat distribution problem. To do so, we first let $\Omega = [0, 1] \times [0, 1]$ with Dirichlet boundary conditions $T = g$ on $\partial\Omega$ and let $r = 1$. We start by discretising the solid Ω to end up with a grid $\mathbb{G}$ and some boundary $\partial\mathbb{G}$.The grid's size thus becomes $(M+1) \times (N+1)$. Using second-order central differences in the directions $\vec{d_1}$ and $\vec{d_2}$, we get

$$-a\frac{U_{m-1}^n - 2U_m^n + U_{m+1}^n}{h^2} - \frac{U_{m-1}^{n-1} - 2U_m^n + U_{m+1}^{n+1}}{h^2} = f_m^n \tag{2}$$

We then introduce the notation $U_P := U_m^n$. Then points in the scheme then become $U_W := U_{m-1}^n$, $U_E := U_{m+1}^n$, $U_{SW} := U_{m-1}^{n-1}$, and $U_{NE} := U_{m+1}^{n+1}$, where E stand for East and describes the solution to grid point east for P, W for West and describes the solution to the grid point west for P and so on. The scheme (2) then becomes

$$-a\frac{U_W - 2U_P + U_E}{h^2} - \frac{U_{SW} - 2U_P + U_{NE}}{h^2} = f_P \tag{3}$$

When solving the system, we represent the 2-dimensional grid as a 1-dimensional vector such that $U_P = U_m^n = U_{n(N+1)+m}$. Thus, writing the scheme in the form of vectors and matrices, we end up with

$$A_h \vec{U} = \vec{b},$$

which is the simplified form for the following representation

$$-\frac{1}{h^2}\begin{bmatrix} -2-2a & a & 0 & \cdots & 1 & \cdots & 0 \\ a & -2-2a & a & \ddots & & \ddots & \vdots \\ 0 & a & -2-2a & \ddots & \ddots & & 2 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 1 & & \ddots & \ddots & -2-2a & a & 0 \\ \vdots & \ddots & & \ddots & a & -2-2a & a \\ 0 & \cdots & 1 & \cdots & 0 & a & -2-2a \end{bmatrix}\begin{bmatrix} U_0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ U_{(n+1)^2-1} \end{bmatrix} = \begin{bmatrix} b_0 \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ b_{(n+1)^2-1} \end{bmatrix}$$

The diagonals that are not a part of the tridiagonal matrices represent the NE and SW points. They are $N+2$ points to the left or right of the diagonal element. If some point $U_P \in \partial\mathbb{G}$, then $b_P = g(P)$. g is our boundary conditions. Furthermore, the P-th row of A_h is zero except for the elements along it diagonal which is -1. If U_P does not lie at the boundary, then $b_P = f(P)$.

After applying this scheme to solve the problem numerically, we want to test the solver on the test problem with

$$f(x,y) := -2(a+1)\cos(5y) + 20x\sin(5y) + 25x^2\cos(5y) \tag{4}$$

as the right-hand side, and

$$g(x,y) = x^2\cos(5y) \tag{5}$$

as the boundary condition. The exact solution to the problem becomes

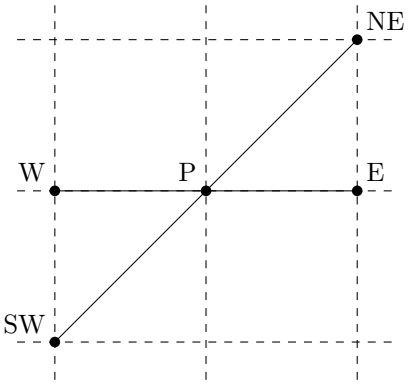
$$u(x,y) = x^2\cos(5y).$$

In the appendix, one can find the plots for the exact and numerical solutions for $r = 1, M = N = 100,$ and $a = 2$ as well as the error plot (fig.2). As we can see, the numerical and analytical solutions seem to be exactly the same with no noticeable differences. The error plot suggests the same as we get $\|\vec{e}\|_\infty < 0.0001$. At the boundaries, the error is 0. This is because we have defined the points at the boundary to be equal to g, which is the exact solution at those points.

Analysis of the derived scheme

We now want to perform an analysis of our scheme to understand its behaviour. To do so, we will show that the scheme has positive coefficients, check whether it is L^∞ stable and find an error bound – all while using the methods we have learned in the course.

Stencil: Given the scheme that we have previously derived (3), its stencil can be illustrated as



Positivity of the coefficients: Since having positive coefficients is crucial for a scheme to be stable, we now

check whether it has positive coefficients. To do so, recall the scheme from (2)

$$-\mathcal{L}_h U_P = \frac{1}{h^2}(-aU^n_{m-1} - aU^n_{m+1} - U^{n-1}_{m-1} - U^{n+1}_{m+1} + (2a+2)U^n_m) \tag{6}$$

$$= \frac{1}{h^2}(-aU_W - aU_E - U_{SW} - aU_{NE} + (2a+2)U_P) \tag{7}$$

$$= \alpha_{P,P}U_P - \sum_{\substack{Q\in S \\ Q\neq P}} \alpha_{P,Q}U_Q, \quad \text{where S} = \{\text{W, E, SW, NE}\} \tag{8}$$

For a scheme to have positive coefficients, it needs to fulfil the following conditions

$$(1) \quad \alpha_{P,Q} \geq 0 \quad Q \in S, Q \neq P \tag{9}$$

$$(2) \quad \alpha_{P,P} \geq \sum_{\substack{Q\in S \\ Q\neq P}} \alpha_{P,Q} \tag{10}$$

Let us start by checking whether (9) is met. We can see from the equation for the scheme (7) that

$$\alpha_{P,W} = \alpha P, E = \frac{a}{h^2}$$

Since both $a, h > 0$ per definition, clearly $\alpha_{P,W} = \alpha P, E > 0$. Additionally, without any need for proof,

$$\alpha_{P,SW} = \alpha P, NE = \frac{1}{h^2} > 0$$

Thus, condition (1) is met. It remains to show that condition (2), i.e. (10), is met to show that the scheme includes positive coefficients only.

$$\alpha_{P,P} = \frac{2a+2}{h^2}$$

$$\sum_{\substack{Q\in S \\ Q\neq P}} \alpha_{P,Q} = \frac{a}{h^2} + \frac{a}{h^2} + \frac{1}{h^2} + \frac{1}{h^2} = \frac{2a+2}{h^2}$$

Thus, $\alpha_{P,P} = \sum_{\substack{Q\in S \\ Q\neq P}} \alpha_{P,Q}$, and the condition holds. Therefore, we have a scheme with positive coefficients.

Now that we have shown that the scheme contains positive coefficients only, we can show that it is stable in L^∞ as well.

Stability in L^∞: Given a supersolution $\phi \geq 0$ on $[0,1]$x$[0,1]$, and using the discrete maximum principle, we want to show that the method is stable in L^∞. Now to be able to use

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

as a supersolution, we want to first be sure that taking the linear differentiating operator on ϕ will yield us a result equal to or above 1. In other words, we want that

$$-\mathcal{L}_h\phi \geq 1 \quad \text{in} \quad \mathbb{G}$$

$$\Rightarrow -\mathcal{L}_h\phi(x) = \frac{1}{h^2}\left(a\delta_x^2 + (\vec{d_2} \cdot \begin{pmatrix}\delta_x \\ \delta_y\end{pmatrix})^2\right)\phi(x) \overset{?}{\geq} 1$$

where $\vec{d_2} = (1,1)$ Since $\phi(x)$ is dependent on x only, differentiating the supersolution in the y-direction will result in 0. Thus,

$$-\mathcal{L}_h\phi = -(a+1)\frac{\phi(x+h) - 2\phi(x) + \phi(x-h)}{h^2}$$

$$= -(a+1)\frac{0.5(x+h) - 0.5(x^2+2hx+h^2) - (x-x^2) + 0.5(x-h) - 0.5(x^2-2hx+h^2)}{h^2}$$

$$= a+1 \geq 1$$

We can therefore use $\phi(x)$ as a supersolution. Define now

$$V_P := \begin{cases} U_P - \|f\|_\infty \phi_P & P \in \mathbb{G} \\ g_P - \|f\|_\infty \phi_P & P \in \partial\mathbb{G} \end{cases}$$

Taking the linear differentiating operator on out newly defined V_p, we get

$$\begin{cases} -\mathcal{L}_h V_P = -\mathcal{L}_h U_P - \|f\|_\infty(-\mathcal{L}_h \phi_P) \\ -\mathcal{L}_h U_P = -f_P \\ -\mathcal{L}_h \phi_P \geq 1 \end{cases} \Rightarrow -\mathcal{L}_h V_P \leq f_P - \|f\|_\infty \leq 0, \quad P \in \mathbb{G}$$

From the discrete maximum principle, we know that the maximum value for U_P must either lie on the boundary of our grid $\partial\mathbb{G}$ or is equal to 0. Thus,

$$\begin{aligned} V_P := U_P - \|f\|_\infty \phi_P &\leq \max_{P\in\partial\mathbb{G}} V_P \leq \max_{P\in\partial\mathbb{G}} |g_P| \\ \iff U_P &\leq \max_{P\in\partial\mathbb{G}} |g_P| + \|f\|_\infty \phi \\ \iff \max_{P\in\mathbb{G}} U_P &\leq \max_{P\in\partial\mathbb{G}} |g_P| + \max_{P\in\mathbb{G}} \phi_P \|f\|_\infty \end{aligned}$$

Letting $g = 0$, and with $\max_{P\in\mathbb{G}} \phi_P = \phi(\frac{1}{2}) = \frac{1}{8}$, where $\frac{1}{2} \in [0, 1]$, we get

$$U_P \leq \frac{1}{8}\|f\|_\infty, \tag{11}$$

showing us that given a change on the right-hand side of the equation, we will always have an upper bound for our solution in the infinity-norm. This means that our scheme is stable in L^∞.

Error bound: Lastly, we want to show that the error term for the scheme is bounded. The error of a numerical solution is defined as

$$e_P = u_P - U_P, \quad P \in \mathbb{G} \tag{12}$$

Taking the discrete linear difference operator on the error term (12), we get

$$-\mathcal{L}_h e_P = -\mathcal{L}_h u_P - (-\mathcal{L}_h U_P) \tag{13}$$

Define the local truncation error as

$$\tau_P = -\mathcal{L}u_P + \mathcal{L}_h u_P = -\mathcal{L}u_P - (-\mathcal{L}_h u_P) \tag{14}$$

We then get for (13)

$$-\mathcal{L}_h e_P = (-\mathcal{L}u_P - \tau_P) - (-\mathcal{L}_h U_P)$$

Recall that by definition, $-\mathcal{L}_h U_P = f_P, -\mathcal{L}u_P = f_P$. Thus,

$$-\mathcal{L}_h e_P = (-\mathcal{L}u_P - \tau_P) - (-\mathcal{L}_h U_P) = -\tau_P.$$

By definition, the error term on the boundary is 0, meaning that $e_P = 0 \quad \forall P \in \partial\mathbb{G}$. Now using the result from L^∞-stability (11) with $U_P = e$ and $f = -\tau$, we end up with

$$\max_{P\in\mathbb{G}} |\vec{e}| = \|\vec{e}\|_\infty \leq \frac{1}{8} \max_{P\in\mathbb{G}} |\vec{\tau}| = \frac{1}{8}\|\vec{\tau}\|_\infty \tag{15}$$

$$\iff \|\vec{e}\|_\infty \leq \frac{1}{8}\|\vec{\tau}\|_\infty. \tag{16}$$

We have thus found a bound for the error term. The main problem now is that we do not know what the local truncation error τ_P is.

Recall the definition of τ_P (14). The terms $\mathcal{L}u_P, \mathcal{L}_h u_P$ can be written as

$$\mathcal{L}u_P = a\partial_x^2 u_P + (\vec{d_2} \cdot \nabla)^2 u_P \tag{17}$$

$$\mathcal{L}_h u_P = \frac{1}{h^2}\left(a\delta_x^2 + (\vec{d_2} \cdot \begin{pmatrix}\delta_x \\ \delta_y\end{pmatrix})^2\right) u_P \tag{18}$$

To be able to get any useful information from the last term (18), we will need to expand the term using Taylor's expansion theorem. This results in

$$\begin{aligned}
&\frac{1}{h^2}\delta_x^2 u_P = u_{xx} + \frac{h^2}{12}u_{xxxx} + \mathcal{O}(h^3) \\
&\frac{1}{h^2}\left(a\delta_x^2 + (\vec{d_2}\cdot\begin{pmatrix}\delta_x\\ \delta_y\end{pmatrix})^2\right)u_P = (\vec{d_2}\cdot\nabla)^2 u_P + \frac{h^2}{12}(\vec{d_2}\cdot\nabla)^4 u_P + \mathcal{O}(h^3) \\
&\Rightarrow \mathcal{L}_h u_P = a(u_{xx} + \frac{h^2}{12}u^{(4)} + (\vec{d_2}\cdot\nabla)^2 u_P + \frac{h^2}{12}(\vec{d_2}\cdot\nabla)^4 u_P + \mathcal{O}(h^3) \\
&\Rightarrow \tau_P = \frac{h^2}{12}(u_{xxxx} + (\vec{d_2}\cdot\nabla)^4 u_P) + \mathcal{O}(h^3).
\end{aligned}$$

Define $K := \left\|\partial_x^4 u(\xi,\eta)\right\|_{L^\infty(\Omega)} + \left\|(\vec{d_2}\cdot\nabla)^4 u(\xi,\eta)\right\|_{L^\infty(\Omega)}$, where $\xi,\eta \in [0,1]$ are chosen in such way that they maximise K. Hence, the local truncation becomes much easier to read

$$\|\vec{\tau}\|_\infty \leq \frac{1}{12}Kh^2$$

Putting it in the equation for the error bound (16), we get

$$\|\vec{e}\|_\infty \leq \frac{1}{96}Kh^2 \tag{19}$$

Thus, the error term is bounded by a constant and h^2, and its rate of convergence is 2.

Testing the code

Now that we have programmed the scheme and have completed a theoretical analysis of the scheme, we want to check whether the code behaves the way it should in theory. To do so, we make tables of errors and convergence rates and then make a log-log plot of the error. We use the same interesting test problem as earlier, recall (4) and (5). We use $a = 2$, and start with $M = N = 5$, and double M, N for each step in the convergence analysis. This yields the table of errors and convergence rates (table 1) which can be found in the appendix. Looking at the table of errors together with the log-log plot of the error in the appendix (fig.3), we can see that the method for this problem converges with order 2. Recalling the upper bound for the error (19), we can see that the result for the convergence rate is in line with the theoretical rate of convergence 2.

New step sizes

Until now we have used $r = 1$. When we change the value of r the direction of $\vec{d_2}$ changes. If we still want to solves the heat distribution problem for $\Omega = [0,1]\times[0,1]$ the method used in 1a) no longer necessarily provides an accurate solution. This can be solved by introducing a new grid where $h = \frac{1}{M}$ and $k = |r|h$. Now we look at a case where r is an irrational number. Since r is irrational the grid will miss the upper boundary ($y = 1$). Assume we hit the boundary, then we get

$$y = k\cdot N = \frac{|r|N}{M} = 1 \quad \Rightarrow |r| = \frac{M}{N}$$

which is a contradiction since $|r| \in \mathbb{I}$ and $\frac{M}{N} \in \mathbb{Q}$. To overcome this problem, we fatten the boundary. An illustration of this can be found in (fig.1). We extend the grid, and move the upper boundary values of $\partial\Omega$ to the upper boundary of $\partial\mathbb{G}$.
We use the same method as earlier. However, this time we use the newly defined grid. After solving for U we move the upper boundary of $\partial\mathbb{G}$ down to $y = 1$. We use the same test problem as earlier, recall (4) and (5), but with $r = \pi$ and $a = 2$. Now observe the error plot in (fig.4). One can notice that the largest error can be found close to the upper boundary, which is where we had to fatten the boundary $\partial\mathbb{G}$. This makes sense as we miss the upper boundary and therefore the calculated value close to the upper boundary should get a worse approximation. By considering the log-log plot in (fig.5) we see that the numerically estimated convergence rate is close to 1. However, by observing the convergence rates in the table of errors in the appendix (table 2), we see that the convergence rate is non-linear. How accurately the grid hits the boundary varies between the step sizes. There is some "luck" involved in this process. This could be the cause of a not so linear line in the log-log plot.

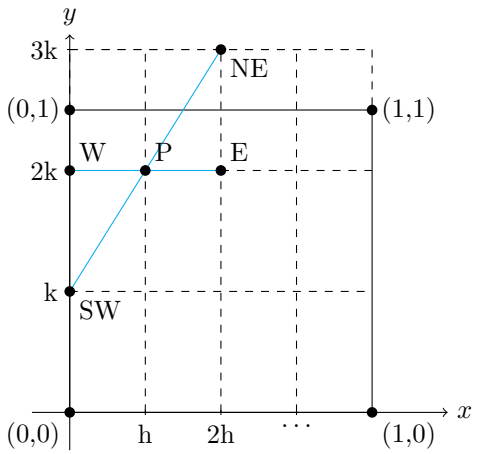


Figure 1: Example of a grid $\mathbb{G}$ after the fattening of the boundary

Methods for solutions for irregular boundaries

In this problem, Ω will once again be the unit square. However, this time it will be enclosed by the curve $y = h(x) := \frac{1}{2}\left(cos(\pi x) + 1\right)$ together with the two axes $(x, y = 0)$. We solve the Dirichlet boundary value problem in the isotropic case when $\kappa = I$ and $\nabla \cdot (\kappa \nabla T) = \Delta T$, where $\Delta = \partial_x^2 + \partial_y^2$. This means that we use the standard central difference stencil, which yields

$$-\mathcal{L}_h U_P = -\frac{U_W + U_E - 4U_P + U_S + U_N}{h^2} = f_P. \tag{20}$$

Like the previous problem we now have a boundary along $h(x)$ that does not necessarily hit the grid points. Two possible solutions are modifying the discretisation near the boundary (MTD) and fattening the boundary (FTB).

We will first consider MTD. For this method we need to introduce $\eta_x := \frac{h^{-1}(y_P) - x_P}{h}$ and $\eta_y := \frac{h(x_P) - y_P}{k}$. If a point P is $\in \Omega$, and $E \notin \Omega$, then η_x describes the relative intersection of $h^{-1}(y_P)$ through the line between P and E. This means that $\eta_x \in (0, 1)$, and if the number is closer to zero, then the intersection is closer to P. And if the number is closer to one, then the intersection is closer to E. η_y works in the same way but for P and N. By modifying the discretisation near the boundary, we end up with an error linear to h. This can be shown as follows: Assume our boundary goes between the points P and N, and P and E respectively. The points where the boundary goes through can be described by E' = P + $(\eta_x h, 0)$ for the new point between E and P, and N' = P + $(0, \eta_y h)$ for the new point between N and P. Now let us for now focus on the error in the x-direction. Now consider the local truncation error in the x-direction.

$$\tau_{x,P} = \partial_x^2 u_P - (a_{E'} u_{E'} + a_P u_P + a_W u_W)$$

where the variables $a_{E'}, a_P, a_W$ are coefficients that comes from taking central differences around u_P. To find the convergence rate of the error, we must find those coefficients. Thus,

$$\begin{aligned}
\partial_x^2 u_P - \tau_{x,P} &= a_{E'} + a_P u_P + a_W u_W \\
&= a_{E'} u(x_P + \eta_x h, y_P) + a_P(x_P, y_P) + a_W u(x_P - h, y_P) \\
&\overset{\text{Taylor}}{=} a_{E'}(u + \eta_1 h u_x + \frac{1}{2}(h\eta_x)^2 u_{xx} + \mathcal{O}(h^3)) + a_P u_P + a_W(u - h u_x + \frac{1}{2}h^2 u_{xx} + \mathcal{O}(h^3))
\end{aligned}$$

From the equation above, we can see that

$$\begin{cases} h(\eta_x a_{E'} - a_W) = 0 \\ \frac{1}{2}h^2(\eta_x^2 a_{E'} + a_W) = 0 \\ \tau_{x,P} = a_{E'} \cdot \mathcal{O}(h^3) + a_W \cdot \mathcal{O}(h^3) \end{cases} \Rightarrow a_{E'} = \frac{2}{h^2 \eta_x(\eta_x + 1)}, \quad a_W = \frac{2}{h^2(\eta_x + 1)}$$

Thus, the local truncation becomes

$$\tau_{x,P} = \frac{2}{h^2 \eta_x(\eta_x + 1)}\mathcal{O}(h^3) + \frac{2}{h^2(\eta_x + 1)}\mathcal{O}(h^3) = \mathcal{O}(h)$$

The same result for the local truncation error can be obtained for the y-direction. Since the error is directly proportional to the local truncation error, i.e. $e_P \propto \tau_P$, we can conclude that the error will be linear, i.e.

$$e_P = \mathcal{O}(h) \tag{21}$$

When we implement the method in our code, the discretisation we use along the boundary is as follows

$$-\frac{2}{h^2}\left(\frac{U_W}{\eta_x+1} - \frac{U_P}{\eta_x} + \frac{U_E}{\eta_x(\eta_x+1)}\right) - \frac{2}{k^2}\left(\frac{U_S}{\eta_y+1} - \frac{U_P}{\eta_y} + \frac{U_N}{\eta_y(\eta_y+1)}\right) = f_P.$$

After we solve the system of equations for U, we set all the values outside our domain of interest to 0 in order to get the correct plot.

Now consider the second method FTB. For each grid point, P contained by $x = 0$, $y = 0$ and $h(x)$ we check if N and E are outside our domain of interest. If E is outside, we assign U_N to be the projected value at h along the y-direction. And similarly for E in the x-direction. Just as in the previous method, the error for FTB is linear. Recall the definition of the error term (16). Denote by $\pi(P)$ the projection of a point P onto $\partial\mathbb{G}$. Since u_P is the value of the projection of a point P outside of $\partial\mathbb{G}$ onto $\partial\mathbb{G}$ itself, we can write $u_P = u_{\pi(P)}$. Thus, the error bound becomes

$$\begin{cases} e_P = u_P - U_P = u_{\pi(P)} + (P - \pi(P))u_x(\xi) - g(\pi(P)) \\ u_{\pi(P)} = g(\pi(P)) \\ P - \pi(P) = \mathcal{O}(h) \end{cases} \Rightarrow e_P = \mathcal{O}(h).$$

Results for solutions for irregular boundaries

Let us now look at the results for both methods. We still use the same interesting test problem as earlier, recall (4) and (5). We have plotted the error for both methods, using $M = N = 100$. Consider the plot for the error in the MTD method (fig.6) and the plot for the error in the FTB method (fig.9) that can be found in the appendix. We observe that the error is largest close to the boundary for both the FTB and the MTD methods. For the FTB method, this is expected for the same reasons as in 1d). Since we get the boundary values outside the domain by projecting the point onto h, the value is not exact and a slight error is expected. However, the error close to the boundary for the MTD method was not expected. In this method, we used the exact values from the boundary conditions, so we expected an error function where the largest error was not so close to the boundary. This may be a consequence of an error in our code.

Now take a look at the log-log plots of the convergence for MTD (fig.7) and FTB (fig.9) in the appendix. We start the convergence analysis with $M = N = 10$ and then double the value for each step. The numerically estimated rate for MTD is $p = 0.85$ and for FTB is $p = 0.84$. Both methods seem to converge at approximately the same rate, which is expected. However, the convergence rate was lower than the expected $p = 1$ for both methods.

Consider the tables of errors for both methods, that is, table 3 for MTD and table 4 for FTB in the appendix. One can observe that the error for the MTD is slightly smaller than for FTB for the same M, N. Although both methods seem to yield about the same accuracy, the FTB method was easier to implement. Furthermore, we did not notice any big differences in computational time.

Conclusion

In conclusion, we developed and analyzed numerical schemes to solve the heat distribution problem in anisotropic and isotropic materials using finite difference methods. Our results showed high accuracy in anisotropic settings, validating the effectiveness of our numerical approach. In handling irregular boundaries, we explored two methods; modifying the discretization of the boundary and fattening the boundary. While both methods yielded comparable accuracy, fattening the boundary proved simpler to implement.

A Appendix

Part 1a)

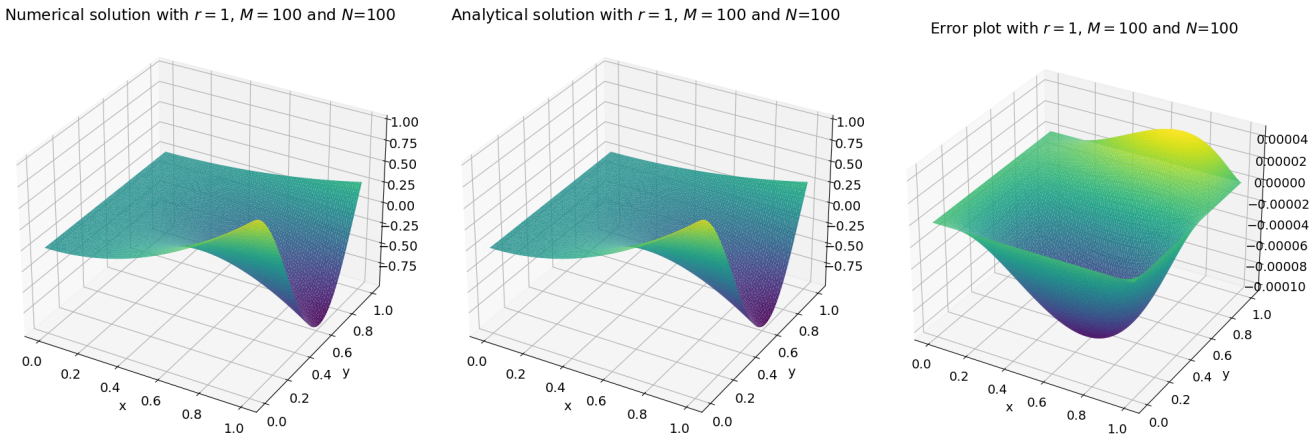


Figure 2: Plots for the numerical and analytical solutions with its error plot for $r = 1$ and $a = 2$.

Part 1c)

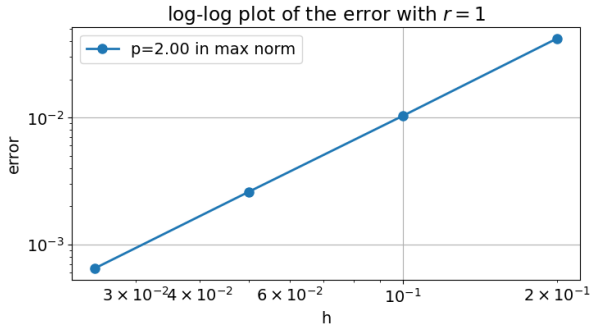


Figure 3: Convergence rate for $r = 1$

h	max. error	rate
0.2	0.04189	0
0.1	0.01032	2.02139
0.05	0.00260	1.98662
0.025	0.00065	2.00337

Table 1: Table of errors for the scheme with $a = 2$ and $r = 1$

Part 1d)

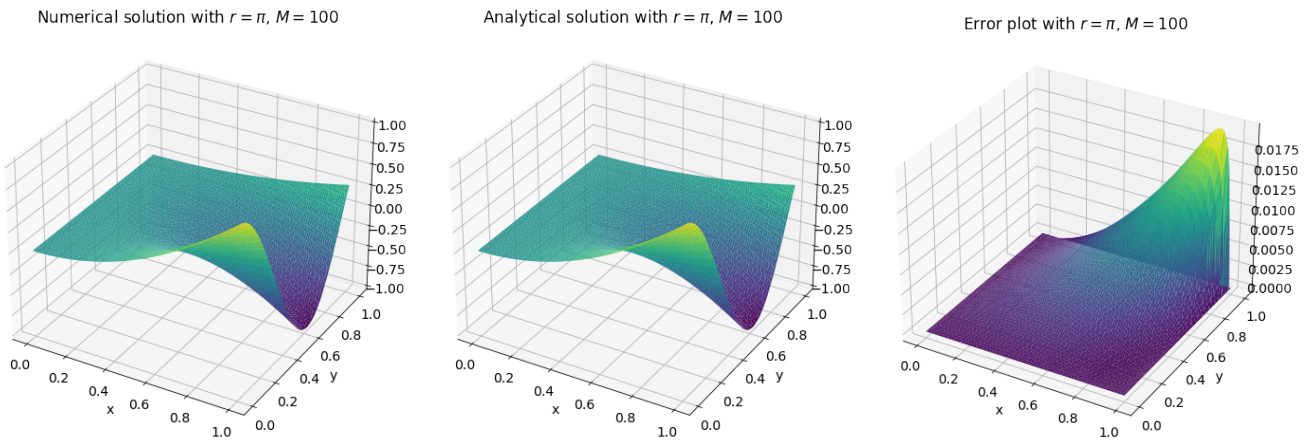


Figure 4: Plots for the numerical and analytical solutions with its error plot for $r = \pi$ and $a = 2$.

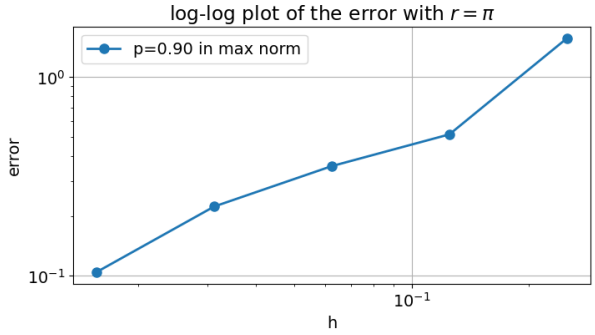


Figure 5: Convergence rate for $r = \pi$

h	max. error	rate
0.2	1.56834	0
0.12500	0.51511	1.60627
0.06250	0.35670	0.53016
0.03125	0.22340	0.67512
0.01562	0.10455	1.0953

Table 2: Table of errors for scheme with $a = 2$ and $r = \pi$

Part 2: MTD

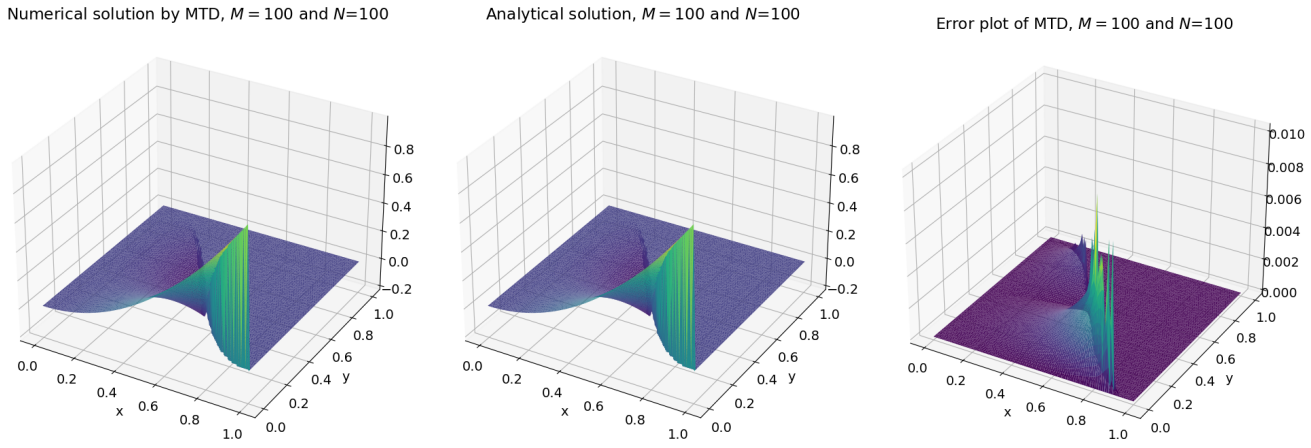


Figure 6: Plots for the numerical and analytical solutions with its error plot for MTD

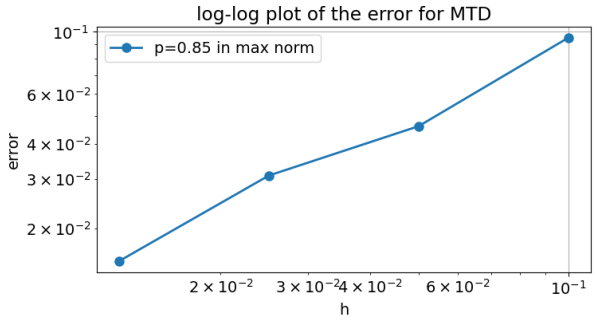


Figure 7: Convergence rate for MTD

h	max. error	rate
0.1	0.09513	0
0.05	0.04611	1.04477
0.025	0.03079	0.58264
0.0125	0.01529	1.01012

Table 3: Table of errors for MTD

Part 2: FTB

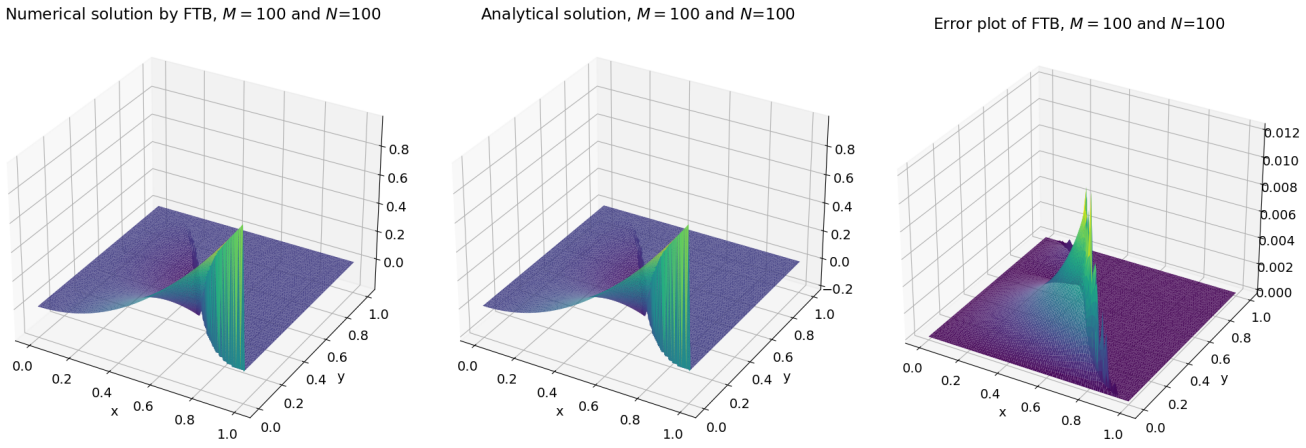


Figure 8: Plots for the numerical and analytical solutions with its error plot for FTB

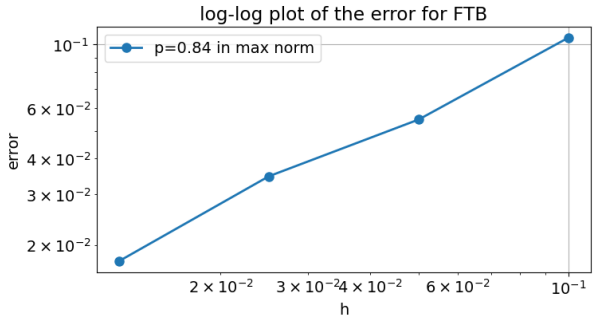


Figure 9: Convergence rate for FTB

h	max. error	rate
0.1	0.10556	0
0.05	0.05475	0.94707
0.025	0.03474	0.65622
0.0125	0.0176	0.98134

Table 4: Table of errors for FTB