MT5846: Advanced Computational Techniques

Project Two

Name: James Portier
Matriculation number: 190009065

Date: April 2024

Abstract

This report presents a comprehensive investigation into iterative methods for solving Laplace's equation, focusing on steady-state heat conduction in a two-dimensional rectangular plate. The study evaluates the efficiency and convergence behavior of five numerical techniques: Jacobi, Point Gauss-Seidel, Line Gauss-Seidel, Point Successive Over-Relaxation (SOR), and Line SOR methods. The investigation encompasses the determination of optimal relaxation parameters (ω) for SOR methods, analysis of initial condition influence on convergence rates, and comparative assessment of convergence behavior across different methods. Numerical experiments reveal that over-relaxation accelerates convergence in both Point and Line SOR methods, with the Line SOR exhibiting the fastest convergence rates. However, instabilities in the Line SOR method underscore the trade-offs between convergence speed and stability. Furthermore, the directional influence of boundary conditions on convergence rates is elucidated through comparative analysis of Line Gauss-Seidel convergence with varying initial conditions. Overall, the study provides valuable insights into the selection and implementation of iterative methods for efficient and stable solution of Laplace's equation in heat conduction problems.

Keywords: Numerical Analysis, Laplace's equation, Steady-State Heat Conduction, Jacobi, Gauss-Seidel, Line Gauss-Seidel, Successive Over-Relaxation (SOR), Line SOR.

Contents

1 Introduction

1.1 The Problem

We consider a flat, two-dimensional rectangular plate for which we aim to calculate the steady-state temperature distribution under a specified set of boundary conditions. The boundary conditions are as follows:

$$\begin{aligned} &\text{At } y = 0, && T = T_1, \\ &\text{At } x = 0, y > 0, && T = T_2, \\ &\text{At } y = H, && T = T_3, \\ &\text{At } x = L, y > 0, && T = T_4. \end{aligned}$$

The rectangular plate has dimensions of one metre in the x direction by two metres in the y direction. Therefore $H = 2$ and $L = 1$. The temperatures, at the boundaries, are given as $T_1 = 100°\text{C}$ and $T_2 = T_3 = T_4 = 0$. The governing PDE for the steady state heat conduction over the metal plate is Laplace's equation,

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0 \tag{1}$$

We take the number of points in the first dimension (x) to be 51 and 101 in the second dimension (y).

1.2 Discretisation and the Five Point Formula

This stencil used in Laplace's model equation (1) gives the difference approximation:

$$\frac{T_{i-1,j} - 2T_{i,j} + T_{i+1,j}}{(\Delta x)^2} + \frac{T_{i,j-1} - 2T_{i,j} + T_{i,j+1}}{(\Delta y)^2} = 0 \tag{2}$$

The second order spatial derivatives are represented by central differences which are second order accurate. One of the most common methods of finite difference formulations is the *Five Point Formula*, illustrated in Figure 1 below:

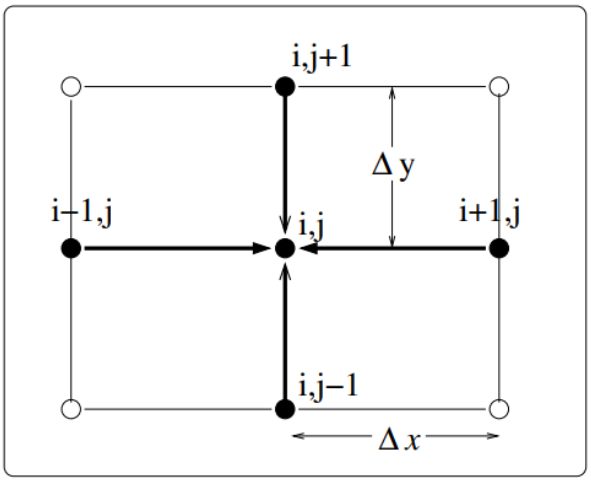

Figure 1: Stencil for the five-point difference formula.

1.3 Methods of Solution

Due to its simplicity we shall consider the five point formula while looking at various methods of solution. The formulation of the solution first involves writing out a linear system of equations, one equation for each grid point. Then we can solve the resulting linear system numerically. Equation (2), the five point stencil approximation for Laplace's equations, can be written as:

$$T_{i-1,j} - 2T_{i,j} + T_{i+1,j} + \left(\frac{\Delta x}{\Delta y}\right)^2 (T_{i,j-1} - 2T_{i,j} + T_{i,j+1}) = 0.$$

Defining β to be the ratio of the step sizes, $\frac{\Delta x}{\Delta y}$, and rearranging gives

$$T_{i+1,j} + T_{i-1,j} + \beta^2 T_{i,j+1} + \beta^2 T_{i,j-1} - 2(1+\beta^2)T_{i,j} = 0. \tag{3}$$

The iterative index will be taken to be k. Note that we are not solving a 'time marching' [1] problem here but are iterating to a steady-state solution. Initially an arbitrary guess at the solution is needed to start the iterative process. For an *initial guess* at $k = 0$, we set $T = 0$ for all interior points. Note that this is not the same as an 'initial condition' - the initial state is a starting point, allowing the method to iterate towards the correct steady state solution.

We now give a brief introduction of each of the methods we'll use to numerically solve the linear system.

1.3.1 Jacobi Method

In this method equation (3) is used to compute the new value $T_{i,j}^{k+1}$ from the old value of its neighbours at iteration, k. The iteration equation can be written down as

$$T_{i,j}^{k+1} = \frac{1}{2(1+\beta^2)}\left[T_{i-1,j}^{k} + T_{i+1,j}^{k} + \beta^2\left(T_{i,j-1}^{k} + T_{i,j+1}^{k}\right)\right] \tag{4}$$

where as mentioned k is the iterative index. The method is convergent if the coefficient matrix is diagonally dominant. Formally put, the sufficient condition for the convergence of the method is that for each row or given i,

$$|a_{i,i}| \geq \sum_{j \neq i} |a_{i,j}|$$

and for at least one row

$$|a_{i,i}| > \sum_{j \neq i} |a_{i,j}|.$$

This is the case for our set-up, and so we expect the method to converge.

[1] *Time marching* in this context refers to iteratively advancing the solution towards a steady state, not in time but through successive approximations until the solution no longer changes significantly.

1.3.2 Point Gauss-Seidel Method

An intuitive improvement on the Jacobi method is make use of the latest approximation for a given point, T^{k+1}, on the computational domain as soon as it becomes available instead of using the T^k value as in the Jacobi method. Assuming the order of calculation is in ascending i and j this gives:

$$T_{i,j}^{k+1} = \frac{1}{2(1+\beta^2)} \left[T_{i-1,j}^{k+1} + T_{i+1,j}^{k} + \beta^2 \left(T_{i,j-1}^{k+1} + T_{i,j+1}^{k} \right) \right]. \tag{5}$$

Similar to the Jacobi method, the method is convergent if the coefficient matrix is diagonally dominant. Again, this is the case for our set-up, and so we expect the method to converge.

1.3.3 Line Gauss-Seidel Method

The *Line Gauss-Seidel* method is an iterative technique for solving partial differential equations that exploits the linear nature of discretized equations. Its formulation involves solving along the grid line by line for constant j. This creates a linear system, to be solved, in turn for each line of constant j in our computational grid. Equation (3), with the unknowns on the LHS becomes,

$$T_{i,j}^{k+1} - 2(1+\beta^2)T_{i,j}^{k+1} + T_{i+1,j}^{k+1} = -\beta^2(T_{i,j-1}^{k+1} + T_{i,j+1}^{k}). \tag{6}$$

Note that in this formulation the boundary values along the line immediately affect the value of the solution to the dependent variable along that line. This implies that if a solution is known to have a larger gradient in one direction it is better to use the Gauss-Seidel method along that direction so as to accelerate convergence. For example, if there is a larger gradient in the horizontal direction, use horizontal lines (for constant j), and if there is a larger gradient in the vertical direction, use vertical lines (for constant i).

1.3.4 Point Successive Over-Relaxation Method (SOR)

Consider the Gauss-Seidel method. The iterative process is converging to the steady state solution, therefore we expect $T_{i,j}^{n}$ to approach $T_{i,j}^{n+1}$. Given this is the case, a good question to ask is can the rate of convergence be speeded up. The *Point Successive Over-Relaxation Method* method attempts to do this, and is formulated as follows:

$$T_{i,j}^{k+1} = (1-\omega)T_{i,j}^{k} + \frac{\omega}{2(1+\beta^2)} \left[T_{i-1,j}^{k+1} + T_{i+1,j}^{k} + \beta^2 \left(T_{i,j-1}^{k+1} + T_{i,j+1}^{k} \right) \right]. \tag{7}$$

If the iterative solution is to converge then $0 < \omega < 2$. If $0 < \omega < 1$ then the process is referred to as *under relaxation*. If $\omega = 1$ then we simply recover the Gauss-Seidel method. If $1 < \omega < 2$ then the process is referred to as *over relaxation*. A natural question to ask is whether there is an optimal value for ω. In general, numerical investigation is required to determine the optimal value. However, in the specific case elliptic PDE's with Dirichlet boundary conditions with constant step sizes (as is the case for our problem), an analytical formula for optimal ω is given as:

$$\omega_{\text{opt}} = \frac{2 - 2\sqrt{1-a}}{a} \tag{8}$$

where

$$a = \frac{\left[\cos\left(\frac{\pi}{im}\right) + \beta^2 \cos\left(\frac{\pi}{jm}\right) \right]^2}{1+\beta^2}.$$

Recall that we took the number of points in x and y to be 51 and 101 respectively. Therefore im = 50 and jm = 100 here. Thus discretisation yields $\beta = \frac{\Delta x}{\Delta y} = \frac{0.02}{0.02} = 1.0$, and so ω_{opt} = 1.9054 (to 5 s.f.).

1.3.5 Line Sucessive Over-Relaxation Method (LSOR)

The line Gauss-Seidel method introduced previously (6) can also be sped up using a relaxation parameter similar to the one used for the point Gauss-Seidel method. The *Line Sucessive Over-Relaxation Method*, with the introduction of the relaxation parameter, is formulated as:

$$\omega T_{i-1,j}^{k+1} - 2(1+\beta^2)T_{i,j}^{k+1} + \omega T_{i+1,j}^{k+1} = -(1-\omega)[2(1+\beta^2)T_{i,j}^{k} - \omega\beta^2(T_{i,j-1}^{k+1} + T_{i,j+1}^{k})]. \quad (9)$$

1.4 Implementation details

1.4.1 Convergence

We use a twofold stopping criterion: stop iterating when convergence has been achieved to satisfactory accuracy, or when a maximum number of 10000 iterations has been reached. For the convergence criterion, we use a measure of difference between the internal points at two successive iterations:

$$\sum_{i=1}^{im-1}\sum_{j=1}^{jm-1}\left|T_{ij}^{k+1} - T_{ij}^{k}\right| < \varepsilon \quad (10)$$

where $\varepsilon = (im-1)(jm-1) \times 1.3 \times 10^{-5}$. This will give a consistent error per point as the spatial resolution changes for different grid sizes.

1.4.2 Banded Solver

The `solve_banded` function from SciPy is optimised for solving linear systems with banded coefficient matrices and offers several advantages over the general `solve` function. It provides memory efficiency by requiring less memory, as only the non-zero diagonal bands need storage. In terms of computational speed, algorithms for banded matrices exploit the sparse structure for faster computations. Algorithmic optimisation is achieved through specialised methods, such as the LU decomposition for banded matrices, which are simpler and more efficient. Additionally, improved stability is ensured as tailored algorithms can offer better numerical stability by minimising operations prone to round-off errors. Since both Line Gauss-Seidel and Line SOR methods involve tri-diagonal systems, we utilise the `solve_banded` function for these methods to capitalise on its enhanced performance and efficiency.

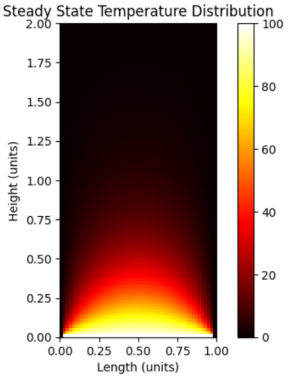

Figure 2: Steady state temperature distribution of the simulation.

2 Results

In this section, we explore and discuss various experiments including an investigation of the optimal relaxation factor ω in both point and line SOR methods, examining the impact of initial conditions on the efficiency of the Line Gauss-Seidel method, and comparing the convergence rates of all presented numerical methods. First, we present the results of the number of iterations required for convergence for each of the numerical methods introduced in the previous section:

Numerical method	**Jacobi**	**Gauss-Seidel**	**Line Gauss-Seidel**	**SOR**	**LSOR**
Number of iterations	5240	2878	1570	121	78

Table 1: Number of iterations required to achieve convergence for each method.

The table summarises the convergence efficiency of various iterative methods, with the LSOR method requiring the least number of iterations, followed closely by SOR, then Line Gauss-Seidel, Gauss-Seidel, and Jacobi methods in increasing order. These results are indicative of the effectiveness of over-relaxation strategies in enhancing convergence rates, especially when applied in a line-wise fashion as in LSOR. Note that for SOR and LSOR we used their theoretically optimal and numerically determined optimal ω values respectively.

2.1 SOR Optimal Omega

In this experiment we demonstrate that the theoretical optimal value of ω is a reasonable approximation by using numerical experimentation. We achieve this by running the point SOR function several times with varying values of ω, and producing a graph of the number of iterations required for convergence for each of these values. We also plot the theoretical optimum, using a red marker. We trial ω values between the range $1 < \omega < 2$, since values below this range wouldn't be considered successive **over** relaxation, and values ≥ 2 **would cause the iterative solution to diverge.**

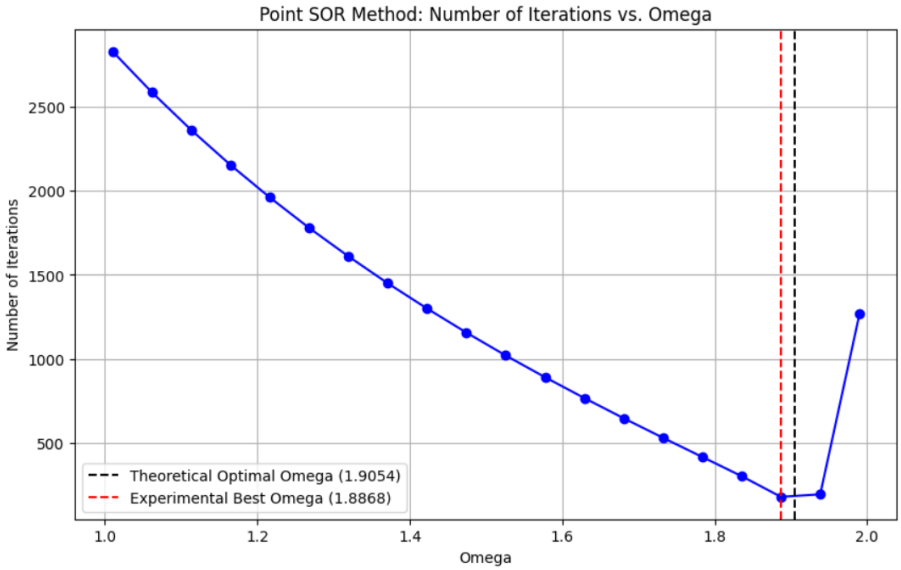

Figure 3: Plot of the number of iterations against varying values of omega, along with its theoretical optimal value.

Figure 3 illustrates the relationship between the relaxation factor ω and the number of iterations required for convergence using the Point SOR method. The theoretical optimal ω of 1.9054, marked by a black dashed line, lies within the experimentally determined effective range. Notably, the numerically calculated optimal ω, indicated by a red dashed line, is 1.8868, demonstrating a close approximation to the theoretical value. Although a finer grid of ω values might result in a more precise match, computational constraints necessitated the current selection. These findings substantiate the theoretical prediction of the optimal ω for the Point SOR method, affirming its practical applicability in achieving efficient convergence.

2.2 LSOR Optimal Omega

Unlike the Point SOR method, there is no analytical formula to calculate the optimal value of ω for the LSOR method. Therefore, our second experiment involves deducing this optimal value through numerical experimentation. During trials with a range of ω values, it was observed that the LSOR method encounters computational issues for ω values greater than approximately 1.34, resulting in errors such as 'array must not contain infs or NaNs'. This phenomenon can be attributed to the nature of relaxation methods, where ω values significantly greater than 1 lead to an over-relaxation process. This over-relaxation amplifies the errors in successive iterations rather than dampening them, eventually causing the numerical solution to diverge rapidly. The divergence manifests in the form of infinite or 'NaN' values within the computation, halting the iterative process. In response to this, we add a try-catch statement to the LSOR method to handle potential numerical instabilities.

To determine the optimal value of ω for the LSOR method, we employ a two-step approach:

1. Initially, we explore a broad range of ω values within $1 < \omega < 1.36$, based on computational stability considerations highlighted previously. This step aims to identify a preliminary estimate of the optimal ω value.

2. Subsequently, we refine our search by focusing on a narrower range around the estimated optimal value from the first step. This phase employs a denser grid of ω values to ensure the accuracy of the optimal value determination, thereby minimising the risk of overlooking the true optimum.

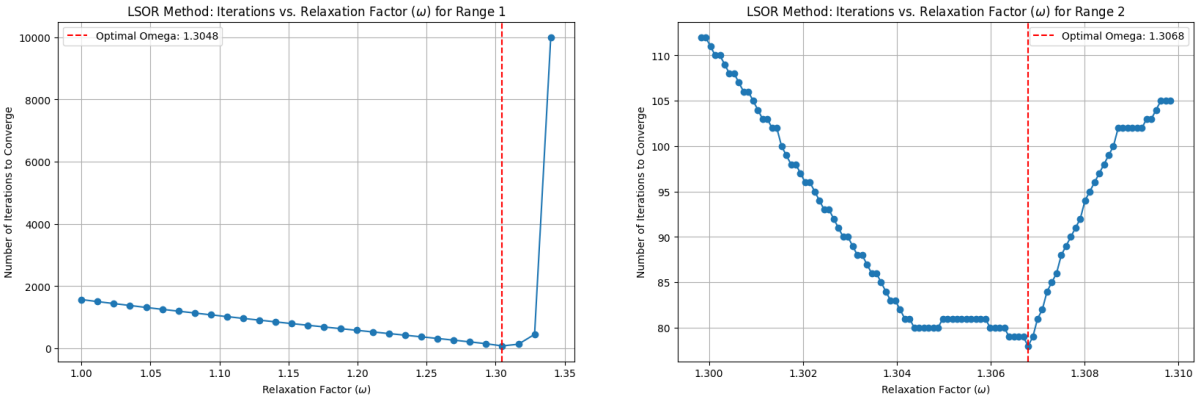

Figure 4: Plot of LSOR method iterations for varying values of omega.

The image comprises two subplots displaying the convergence behavior of the LSOR method. On the left, the first plot shows the number of iterations to convergence for a broad range of ω from 1 to 1.34, with a notable spike as ω approaches 1.34. The optimal ω value, denoted by a red dashed vertical line,

is at 1.3048, which coincides with the minimum number of iterations required. The subplot on the right provides a zoomed-in view for ω values between 1.300 and 1.310, revealing a detailed undulation in the iteration count and marking the optimal ω at **1.3068** with a similar red dashed line, signifying the most efficient convergence within the scrutinised range.

2.3 Influence of Initial Conditions on Line Gauss-Seidel Convergence

For our third experiment, we assess the impact of initial guesses on the convergence rate of the Line Gauss-Seidel method. By initializing the temperature T to 50 for all interior points and employing both lines of constant j and constant i (see 1.3.3) for the linear systems, we observed the iterative behavior. With lines of constant j, the convergence was reached after **1881** iterations, whereas utilizing lines of constant i resulted in convergence after **1873** iterations. This suggests a marginal difference in efficiency between the two approaches, with vertical lines (constant i) demonstrating a slightly faster convergence for this specific initial guess. The observed disparity in convergence rates can be attributed to the directional influence of boundary conditions in the Line Gauss-Seidel method. Since the boundary values along a line directly affect the solution along that line, convergence is accelerated when the method is aligned with the direction of the steepest temperature gradient. Therefore, the marginally faster convergence using lines of constant i can be attributed to the slightly steeper temperature gradient in the vertical direction for the initial guess of $T = 50$.

2.4 Comparing the rates of Convergence

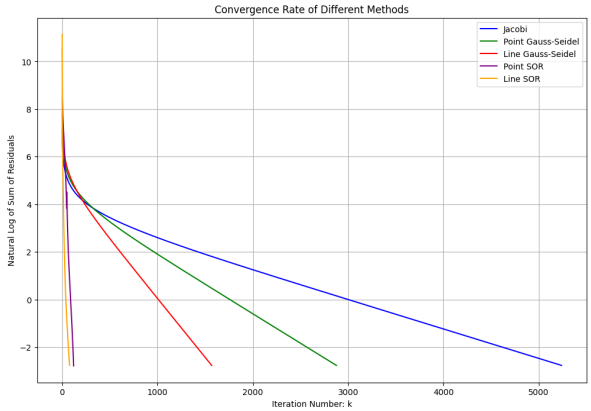

Figure 5: Combined plot of the rate of convergence for each method.

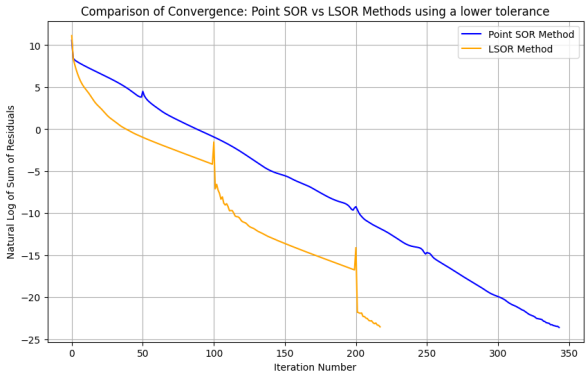

Figure 6: Point SOR and LSOR convergence with lower tolerance.

Figure 5 portrays the natural logarithm of the sum of residuals against the iteration number for each of our numerical methods. From the plot, we observe that the Jacobi method exhibits the slowest rate of convergence among the methods, as indicated by its relatively higher residual values that decrease more gradually. The Point Gauss-Seidel and Line Gauss-Seidel methods demonstrate an improved rate of convergence, with the Line Gauss-Seidel slightly outperforming the Point Gauss-Seidel, evident from their steeper slopes and thus faster drop in residual values. The Point SOR and Line SOR methods show the most rapid convergence rates, with the Line SOR having the steepest initial descent, suggesting that it quickly reaches a solution close to the steady state. Overall, the Point and Line SOR methods converge faster than the classical iterative methods, Jacobi and Gauss-Seidel, with the Line SOR showing a particularly sharp reduction in the sum of residuals, indicating a swift approach to the steady-state solution.

Discerning the differences between the Point SOR and Line SOR methods from Figure 5 is challenging, thus warranting a more detailed analysis. For this purpose, the tolerance was decreased, and a closer look at these methods is presented in Figure 6. The LSOR method is observed to have the most substantial rate of change in the log sum residuals, signifying the fastest convergence. This acceleration is a consequence of the LSOR's line-to-line consideration, allowing for larger iteration steps compared to the Point SOR's point-to-point approach. Despite its rapid convergence, the LSOR method exhibits instabilities, particularly noticeable around iterations 100 and 200, which align with multiples of the grid's vertical dimension ($im + 1 = 101$). Such instabilities are inherent to SOR schemes due to their dependency on the previous iteration's line state. As the method advances through the iterations, cumulative errors may induce peaks in instability, which are then corrected, resulting in noticeable decreases in the log sum of residuals. Conversely, the Point SOR method displays similar, yet less pronounced, instability at iterations 50, 100, and 250. The propagation of errors is less severe due to the point-wise calculation, and the logarithmic measurement scale further dampens the perceived impact. To further investigate these phenomena, a Von Neumann stability analysis and a re-evaluation of the spatial discretisation strategy are recommended.

3 Discussion and Conclusion

The numerical experiments conducted provide valuable insights into the behavior and efficiency of various iterative methods for solving Laplace's equation governing steady-state heat conduction. The results highlight the significance of selecting appropriate relaxation parameters, initial conditions, and computational strategies to achieve optimal convergence rates.

The data presented in the convergence Table 1 clearly indicate the superior efficiency of the SOR and LSOR methods when employing their respective optimal ω values. It is crucial to note that these methods outperformed the Jacobi, Gauss-Seidel, and Line Gauss-Seidel methods under these specific conditions. Had other values of ω been used, the convergence rates for SOR and LSOR might not have been as competitive.

In the investigation of relaxation methods, the determination of the optimal relaxation parameter (ω) emerges as a critical factor influencing convergence efficiency. The experiments demonstrate that while the theoretical predictions of optimal ω values offer valuable guidance, numerical experimentation remains essential to validate and refine these estimates. The observed convergence behavior of the Point SOR and LSOR methods underscores the effectiveness of over-relaxation in accelerating iterative convergence, particularly evident in LSOR's rapid rate of convergence. However, the occurrence of instabilities in LSOR emphasizes the importance of understanding and mitigating the inherent trade-offs associated with relaxation methods, such as balancing convergence speed and stability.

Moreover, the comparative analysis of initial condition influence on Line Gauss-Seidel convergence reveals the nuanced impact of boundary conditions on iterative solutions. The marginal difference in convergence rates between lines of constant j and constant i underscores the directional influence of boundary conditions on the convergence behavior, necessitating careful consideration during method selection and implementation.

Overall, the study underscores the iterative nature of numerical methods for solving Laplace's equation and emphasises the importance of methodological considerations, such as relaxation parameter selection, initial condition specification, and computational strategies, in achieving efficient and stable convergence. The insights gained from these experiments contribute to a deeper understanding of iterative solution techniques and provide valuable guidance for their practical implementation in solving complex heat conduction problems.

References

[1] Irene Kyaza *MT5846 Lecture Notes*. University of St. Andrew's School of Mathematics and Statistics