Solution Methods in Numerical Linear Algebra

Direct and Iterative Solvers for Partial Differential Equations

J.T. den Hertog

Dutch title:
Oplossingsmethoden in Numerieke Lineare Algebra

TU Delft

Solution Methods in Numerical Linear Algebra

Direct and Iterative Solvers for Partial Differential Equations

by

J.T. den Hertog

to obtain the degree of Bachelor of Science
at the Delft University of Technology,
to be defended publicly on Monday June 23, 2025 at 10:00 AM.

Student number:	5827264	
Project duration:	April 22, 2025 – June 16, 2025	
Thesis committee:	Dr. C.A. Urzúa-Torres,	Supervisor
	A.C. Wisse, MSc,	Supervisor
	Dr. T.W.C. Vroegrijk	

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

This thesis was written as part of the Bachelor's programme in Applied Mathematics at Delft University of Technology. It presents the results of my research on the behaviour of direct and iterative solvers for systems arising from partial differential equations.

First and foremost, I thank God for the strength, and perseverance that made this work possible.

I would also like to express my sincere gratitude to dr. Carolina Urzúa-Torres and Anouk Wisse MSc. for their supervision, insightful feedback, and support throughout this project. Their guidance was invaluable.

Jan den Hertog
Delft, June 2025

Summary

Laymen's summary

Many scientific problems, from predicting heat flow to simulating waves, are described by equations that can't be solved exactly. Instead, we approximate the solutions using a computer, which turns these problems into systems of linear equations. Solving these systems quickly and accurately is essential in engineering, science, and data analysis.

This thesis looked at how the setup of these equations affects the performance of the methods used to solve them. The equations depend on the physical situation, like whether a material is insulated or fixed at the ends. These choices change the structure of the resulting equations, sometimes making them easier or harder to solve.

We studied two main types of solution methods:

- Direct methods, which try to solve the problem in one shot (like solving for x in $a \cdot x = b$).
- Iterative methods, which guess the solution and improve the guess step-by-step.

We found that direct methods are reliable when the system is "well-behaved" (non-singular), but can fail or become unstable when it is close to being unsolvable. Iterative methods are more flexible, and surprisingly, they sometimes still work even when theory says they should not. However, their success depends on mathematical properties, like how dominant the diagonal values in the matrices are.

We also saw that if the system becomes "almost unsolvable" (for example, due to the size of the setup), both types of methods can break down or become very slow. This shows that being careful in how we model and discretize physical problems is just as important as choosing the right solver.

In short, the way we write the equations strongly influences how well computers can solve them, and understanding this link helps us design faster and more reliable simulations.

Summary

This thesis investigated the influence of matrix structure, arising from various boundary conditions, on the performance of both direct and iterative solvers for linear systems resulting from discretized partial differential equations (PDEs). Using the negative one-dimensional Poisson equation as a model problem, system matrices were constructed for Dirichlet, Neumann, and mixed boundary conditions, including symmetrized variants.

The study classified the resulting matrices in terms of singularity and positive definiteness. Direct solvers such as LU decomposition and Cholesky factorization were tested for accuracy and applicability. Iterative solvers (Jacobi, Gauss-Seidel, and Successive Over-Relaxation or SOR) were analysed using the spectral radius of their iteration matrices to predict convergence.

Key findings include:

- Matrices from Dirichlet and mixed boundary conditions were non-singular; Neumann matrices were singular but still allowed convergence in iterative solvers under specific conditions.
- Cholesky factorization only applied to positive definite systems, while LU decomposition was more generally applicable to non-singular systems.
- Gauss-Seidel converged faster than Jacobi, with twice the convergence speed in tridiagonal cases.
- SOR achieved the fastest convergence when using the optimal relaxation parameter, though this optimal ω is generally not known a priori.
- Even when the spectral radius $\rho(B)$ approached 1, as in Neumann systems, convergence was observed due to the structure of the eigenvalue spectrum.

The thesis concludes that the structural properties of the matrix, such as singularity and diagonal dominance, are critical in determining the suitability and efficiency of numerical solvers. It also highlights the fragility of these methods when applied to nearly singular systems, motivating future research into more robust solution techniques and convergence criteria.

Contents

1 Introduction

Partial differential equations (PDEs) play a central role in the mathematical modelling of physical, biological, and engineering systems. They describe a wide range of phenomena such as heat conduction, fluid dynamics, electromagnetism, population dynamics, and structural mechanics. However, for most problems, PDEs are too complex to solve analytically, especially when the domain geometry is irregular or the coefficients are variable. In such cases, numerical methods are essential for obtaining approximate solutions [13].

Obtaining the numerical solution of PDEs involves several steps, beginning with the discretization of the spatial and, if applicable, temporal domains. This process transforms the continuous PDE into a system of algebraic equations. The result is often a large system of equations of the form

$$A\mathbf{u} = \mathbf{f}, \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$ is the system matrix resulting from the discretization, $\mathbf{f} \in \mathbb{R}^n$ is a known vector containing source terms and boundary conditions, and $\mathbf{u} \in \mathbb{R}^n$ is the unknown vector approximating the solution of the original PDE. Depending on the discretization method, the matrix A may be sparse or dense. Furthermore, if the original PDE is nonlinear, additional techniques such as Picard iteration or Newton's method may be required to solve the resulting system [15, ch. 6].

In order to obtain accurate approximations, the system size n is typically large. However, the matrix A often has important structural properties such as sparsity, symmetry, and positive definiteness. For instance, the Laplace operator discretized with finite differences yields, depending on the boundary conditions, symmetric sparse matrices with a regular pattern. Despite such favourable properties, directly computing the inverse of A is computationally infeasible for large-scale problems, due to both memory constraints and numerical instability [11].

Consequently, the development of efficient numerical solvers for large linear systems is a key topic in scientific computing. These solvers can be roughly divided into direct methods and iterative methods. Direct solvers eliminate equations through a predetermined sequence of operations, typically producing a highly accurate solution. However, their computational cost and memory requirements grow rapidly with problem size. Iterative methods, on the other hand, generate a sequence of approximate solutions that (ideally) converge to the true solution. Classical iterative solvers like Jacobi, Gauss-Seidel, and Successive Over-Relaxation (SOR) require less computational effort per iteration compared to direct solvers, but their convergence depends heavily on the properties of the matrix A [8, 17].

Over the decades, significant research has been done on improving the efficiency and robustness of solvers for linear systems arising from PDE discretizations. For structured matrices like those resulting from 1D and 2D discretizations of the Laplacian, researchers have developed specialized techniques that exploit properties like symmetry and positive definiteness to improve performance. A well-known example is the use of Cholesky factorization for symmetric positive definite systems [7, ch. 4]. Iterative methods have also advanced considerably, particularly through the development of preconditioners, which aim to accelerate con-

vergence by transforming the system into a more favourable form [3].

Despite these advances, the performance of a solver, both in terms of speed and accuracy, still depends heavily on matrix properties such as symmetry, positive definiteness, and the presence of a non-trivial kernel. In this context, we investigated how these specific structural features influence solver behaviour, and to what extent assumptions like symmetry and positive definiteness are truly necessary for good performance. This was done by systematically comparing different solvers under varying combinations of these properties.

Research Question

This thesis investigates the relationship between the structural properties of system matrices and the performance of direct and iterative solvers. Specifically, we aim to answer the following question:

> *How does the structure of system matrices affect the behaviour of direct and iterative solvers in terms of speed and accuracy?*

The focus is on systems arising from the discretization of the negative one-dimensional Laplacian operator

$$-\Delta \mathbf{u} = \mathbf{f}, \tag{1.2}$$

as this setting provides an environment to investigate solver behavior while maintaining relevance to a wide class of PDEs.

Thesis Outline

This thesis is organized as follows:

Chapter 2: Preliminaries
Provides the essential mathematical background required for understanding the rest of the thesis.

Chapter 3: Matrix Structures from PDE Discretizations
Introduces the matrix structures that result from discretizing the Laplacian in one dimension using finite differences with different types of boundary conditions.

Chapter 4: Direct Methods
Explores direct solution techniques, particularly LU and Cholesky decomposition.

Chapter 5: Iterative Methods
Presents classical iterative solvers including Jacobi, Gauss-Seidel, and Successive Over-Relaxation (SOR). Convergence criteria are discussed, and the influence of matrix properties on convergence speed is analysed.

Chapter 6: Convergence of direct and indirect methods
Presents an analysis of the order of convergence for different solvers and explores the role of the spectral radius in the convergence behaviour of iteration matrices.

Chapter 7: Conclusion
Gives an overview of the observations and provides an answer to the research question.

Chapter 8: Discussion of results and Future Work
Reflects the findings and proposes directions for future research.

2

Preliminaries

This chapter provides the essential mathematical background required for the rest of this thesis. It includes key concepts from linear algebra such as matrix structure, eigenvalues, positive definiteness, and spectral properties.

2.1. Matrix Fundamentals

Definition 2.1 (Nullspace / kernel)**.** The *nullspace / kernel* $N(A)$ of a matrix $A \in \mathbb{R}^{m\times n}$ is given by [6, sec. 1.6]

$$N(A) = \{\mathbf{x} \in \mathbb{R}^n | A\mathbf{x} = \mathbf{0}\}. \tag{2.1}$$

Theorem 2.1. *Let $A \in \mathbb{R}^{n\times n}$ be a square matrix such that the sum of the entries in each row of A is zero. Then A is singular.*

Proof. Let $\mathbf{1} = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}^T \in \mathbb{R}^n$ be the column vector of all ones.

For any row $i \in \{1, \ldots, n\}$, the i-th component of the product $A\mathbf{1}$ is:

$$(A\mathbf{1})_i = \sum_{j=1}^{n} a_{ij}, \tag{2.2}$$

which is zero by assumption. Therefore,

$$A\mathbf{1} = \mathbf{0}. \tag{2.3}$$

This means that $\mathbf{1}$ is a nonzero vector in the null space of A, so the null space is nontrivial and A is not invertible. Hence, A is singular. □

2.2. Spectral Theory

Definition 2.2 (Eigenvalue and Eigenvector)**.** Let $A \in \mathbb{C}^{n\times n}$. A scalar $\lambda \in \mathbb{C}$ is called an *eigenvalue* of A if there exists a nonzero column vector $\mathbf{v} \in \mathbb{C}^n$ such that

$$A\mathbf{v} = \lambda\mathbf{v}. \tag{2.4}$$

The vector $\mathbf{v}$ is then called an *eigenvector* of A corresponding to λ [6, defn. 5.1].

Definition 2.3 (Spectrum)**.** Let $A \in \mathbb{C}^{n\times n}$. The *spectrum* of A is the set of all its eigenvalues, denoted by $\sigma(A)$ [20, defn. 2.2.1b].

Definition 2.4 (Spectral Radius)**.** The *spectral radius* $\rho(A)$ of a matrix $A \in \mathbb{R}^{n\times n}$ is defined as [20, defn. 2.7.1]

$$\rho(A) = \max\{|\lambda| : \lambda \in \sigma(A)\}. \tag{2.5}$$

Definition 2.5 (Similar Matrices)**.** Matrices $A, B \in \mathbb{R}^{n\times n}$ are said to be *similar* if there exists an invertible matrix $P \in \mathbb{R}^{n\times n}$ such that [6, defn. 5.4]

$$A = P^{-1}BP. \tag{2.6}$$

Proposition 2.1. *If matrices A and B are similar, then $\sigma(A) = \sigma(B)$ [16, lem. 1.11b].*

Proof: See proof of Lemma 1.11b of [16].

Theorem 2.2 (Gershgorin circle theorem)**.** *Let $A \in \mathbb{R}^{n\times n}$. The eigenvalues of A lie in the complex plane within the union of the Gershgorin discs [21, thm. 7.3.1]*

$$|z - a_{ii}| \leq \sum_{\substack{j\neq i \\ j=1}}^{n} |a_{ij}|, \quad \text{where } z \in \mathbb{C}. \tag{2.7}$$

Proof. See proof of Theorem 7.3.1 in [21].

Theorem 2.3 (Jordan normal form)**.** *For any matrix $A \in \mathbb{C}^{n\times n}$, there exists a nonsingular matrix $P \in \mathbb{C}^{n\times n}$ and a block-diagonal matrix $J \in \mathbb{C}^{n\times n}$ such that*

$$J = P^{-1}AP, \tag{2.8}$$

where $J = \operatorname{diag}(J_1, \ldots, J_q)$, and each $J_i \in \mathbb{C}^{m_i \times m_i}$ is a Jordan block of the form

$$J_i = \begin{bmatrix} \lambda_i & 1 & & & \\ & \lambda_i & 1 & & \\ & & \ddots & \ddots & \\ & & & \lambda_i & 1 \\ & & & & \lambda_i \end{bmatrix}, \tag{2.9}$$

with $m_1 + \cdots + m_q = n$ [7, thm. 7.1.9].

Proof. See proof of Theorem 3.1.11 in [10].

2.3. Special Matrix Classes

Definition 2.6 (Strictly diagonal dominant)**.** A matrix $A \in \mathbb{R}^{n\times n}$ is *strictly diagonal dominant* if [16, thm. 3.2]

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}| \quad \text{for all } 1 \leq i \leq n. \tag{2.10}$$

Definition 2.7 (Weakly diagonal dominant)**.** A matrix $A \in \mathbb{R}^{n\times n}$ is *weakly diagonal dominant* if [1]

$$|a_{ii}| \geq \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}| \quad \text{for all } 1 \leq i \leq n. \tag{2.11}$$

Definition 2.8 (Positive definite)**.** A matrix $A \in \mathbb{R}^{n\times n}$ is called *positive definite* if [7, sec. 4.2]

$$\mathbf{x}^T A\mathbf{x} > 0 \quad \text{for all } \mathbf{x} \neq \mathbf{0}. \tag{2.12}$$

Definition 2.9 (Positive semidefinite)**.** A matrix $A \in \mathbb{R}^{n\times n}$ is called *positive semidefinite* if [19, defn. 2.4.1]

$$\mathbf{x}^T A\mathbf{x} \geq 0 \quad \text{for all } \mathbf{x} \neq \mathbf{0}. \tag{2.13}$$

Theorem 2.4. *A matrix $A \in \mathbb{R}^{n\times n}$ is positive definite if and only if the symmetric matrix*

$$B = \frac{A + A^T}{2} \tag{2.14}$$

has positive eigenvalues [7, thm. 4.2.3].

Proof. See proof of Theorem 4.2.3 in [7].

2.4. Permutations and Structured Matrices

Definition 2.10 (Permutation matrix)**.** A *permutation matrix* is a square matrix obtained by permuting the rows (or columns) of the identity matrix.
Equivalently, a matrix $P \in \mathbb{R}^{n\times n}$ is a permutation matrix if in every row and every column there is exactly one entry equal to 1, and all other entries are 0 [7, sec. 1.2.8].

Theorem 2.5. *Every permutation matrix $P \in \mathbb{R}^{n\times n}$ is invertible, with inverse $P^{-1} = P^T$ [17, sec. 3.3.1].*

Lemma 2.1. *The matrix $P \in \mathbb{R}^{n\times n}$ defined by*

$$p_{i,j} = \begin{cases} 1 & \text{if } 1 \le i \le k \text{ and } j = 2i-1, \\ 1 & \text{if } k+1 \le i \le n \text{ and } j = 2i-2k, \\ 0 & \text{otherwise,} \end{cases} \tag{2.15}$$

with $k = \lceil \frac{n}{2} \rceil$, is a permutation matrix.

Proof. First, we show that each row of P contains exactly one entry equal to 1:

- For $1 \le i \le k$, the value $2i-1$ is an odd number in $\{1,3,5,\dots\}$, which is within bounds because $2k-1 \le n$.
- For $k+1 \le i \le n$, the value $2i-2k = 2(i-k)$ is even and in $\{2,4,\dots,n\}$, again within bounds.

Therefore, in each row exactly one column is assigned a 1.

Now we show that each column contains exactly one entry equal to 1. Let $1 \le j \le n$:

- If j is odd, then $j = 2i-1$ for $i = \frac{j+1}{2} \le k$, and hence $p_{\frac{j+1}{2},j} = 1$.
- If j is even, then $j = 2i-2k$ for $i = k + \frac{j}{2} \le n$, and hence $p_{k+\frac{j}{2},j} = 1$.

As each row and each column has exactly one entry equal to 1, P is a permutation matrix. □

Definition 2.11 (Property A)**.** A matrix T is said to have *property A* if there exists a permutation matrix P such that

$$PTP^T = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix}, \tag{2.16}$$

where T_{11} and T_{22} are diagonal matrices [5, defn. 6.12].

Lemma 2.2. *If $T \in \mathbb{R}^{n\times n}$ is tridiagonal, then T has property A.*

Proof. This proof is based on [5, ex. 6.8].

Let T be a tridiagonal matrix. That means that the nonzero entries are only on the main diagonal, subdiagonal, and superdiagonal.

We need to show that there exists a permutation matrix P such that $PTP^T = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix}$, where T_{11} and T_{22} are diagonal matrices.

We now apply the permutation P from Lemma 2.1. This rearranges the rows and columns such that:

- The rows corresponding to odd-numbered indices (originally indexed as 1, 3, 5, ...) come first.
- The rows corresponding to even-numbered indices (originally indexed as 2, 4, 6, ...) come after.

Similarly, the columns are permuted in the same order, resulting in the matrix PTP^T.

In the new matrix PTP^T, the coupling between the even and odd rows in the tridiagonal structure gets separated into off-diagonal blocks T_{12} and T_{21}, while the diagonal blocks T_{11} and T_{22} correspond to rows/columns

originally only connected to their immediate neighbours of the same parity.

However, since in the original matrix T, odd-numbered rows (and columns) are connected only to even-numbered neighbours and vice versa, the submatrix formed from just odd or just even indices contains only diagonal entries. Therefore, both T_{11} and T_{22} are diagonal matrices.

Hence, this proves that such a permutation exists and that tridiagonal matrices have property A. □

Definition 2.12 (Consistently Ordered Matrix)**.** Let $A \in \mathbb{R}^{n\times n}$, with decomposition $A = L + D + U$, where L is strictly lower triangular, D is diagonal, and U is strictly upper triangular. Define

$$B(\alpha) = -D^{-1}\left(\alpha L + \frac{1}{\alpha}U\right), \quad \alpha \in \mathbb{C}\setminus\{0\}. \tag{2.17}$$

If the eigenvalues of $B(\alpha)$ are independent of α, then A is called a *consistently ordered matrix* [5, defn. 6.14].

Theorem 2.6. *If $T \in \mathbb{R}^{n\times n}$ is tridiagonal, then T is consistently ordered.*

Proof. This proof is adapted from [5, sec. 6.5.5], [18, ch. 4] and [16, ex. 3.2].

Let $T \in \mathbb{R}^{n\times n}$ be tridiagonal. By Lemma 2.2, there exists a permutation matrix P such that

$$PTP^T = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix} = C, \tag{2.18}$$

where T_{11} and T_{22} are diagonal matrices.

Consider the standard splitting $T = L + D + U$, where L is strictly lower triangular, D is diagonal, and U is strictly upper triangular. Define the iteration matrix

$$B_C(\alpha) = -D^{-1}\left(\alpha L + \frac{1}{\alpha}U\right), \tag{2.19}$$

for $\alpha \in \mathbb{C}\setminus\{0\}$.

Given the block form of C, the splitting yields

$$D = \operatorname{diag}(T_{11}, T_{22}), \quad L = \begin{bmatrix} 0 & 0 \\ T_{21} & 0 \end{bmatrix}, \quad U = \begin{bmatrix} 0 & T_{12} \\ 0 & 0 \end{bmatrix}. \tag{2.20}$$

Substituting into the expression for $B_C(\alpha)$, we obtain

$$B_C(\alpha) = -\begin{bmatrix} 0 & \frac{1}{\alpha}T_{11}^{-1}T_{12} \\ \alpha T_{22}^{-1}T_{21} & 0 \end{bmatrix}. \tag{2.21}$$

Consider the similarity transformation with

$$S = \begin{bmatrix} I & 0 \\ 0 & \alpha I \end{bmatrix}, \quad S^{-1} = \begin{bmatrix} I & 0 \\ 0 & \alpha^{-1} I \end{bmatrix}. \tag{2.22}$$

Then

$$S^{-1}B_C(\alpha)S = -\begin{bmatrix} 0 & T_{11}^{-1}T_{12} \\ T_{22}^{-1}T_{21} & 0 \end{bmatrix} = B_C(1). \tag{2.23}$$

Hence, $B_C(\alpha)$ is similar to $B_C(1)$ for all $\alpha \in \mathbb{C}\setminus\{0\}$, and therefore, by Proposition 2.1,

$$\sigma(B_C(\alpha)) = \sigma(B_C(1)) \quad \text{for all } \alpha \in \mathbb{C}\setminus\{0\}. \tag{2.24}$$

Since C is a permutation of the original tridiagonal matrix T, it follows that C and T are similar and thus have the same spectrum. Furthermore, the splitting $C = L + D + U$ transforms correspondingly under the permutation, so the associated iteration matrix $B_T(\alpha)$ is similar to $B_C(\alpha)$. Thus

$$\sigma(B_T(\alpha)) = \sigma(B_C(\alpha)) = \sigma(B_C(1)) \quad \text{for all } \alpha \in \mathbb{C}\setminus\{0\}. \tag{2.25}$$

Therefore, T is consistently ordered.

□

2.5. Error and Accuracy

To assess the quality of a numerical method, we need an objective measure that quantifies how accurately it approximates the exact solution.

One commonly used measure is the root mean square error (RMSE). The RMSE is defined as the square root of the mean of the squared differences between the numerical approximation and the exact solution [2].

The formula for the RMSE is given by

$$\text{RMSE}(\mathbf{x}, \hat{\mathbf{x}}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{x}_i - x_i)^2}, \tag{2.26}$$

where

n is the number of unknowns,
x_i is the exact solution, and
$\hat{x}_i$ is the numerical approximation.

3

Discretization Matrices

This chapter derives the general linear system

$$A\mathbf{u} = \mathbf{f}, \tag{3.1}$$

which results from the discretization of the one-dimensional Poisson equation

$$-\Delta\mathbf{u} = \mathbf{f}. \tag{3.2}$$

The construction of the matrix A is carried out for different types of boundary conditions, leading to systems with varying structural properties.

The 1D-interval $[a,b]$ is split into $n+1$ equidistant subintervals of length $\Delta x = \frac{b-a}{n+1}$ (see also Figure 3.1). The nodes are given by $x_j = a + j\Delta x$, for $j = 0,\ldots,n+1$. The approximation of $u_j = u(x_j)$ is denoted by w_j.

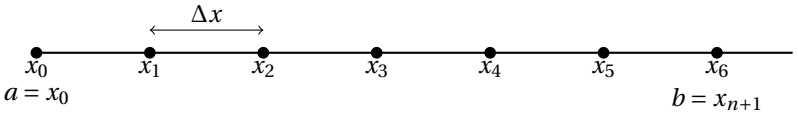


Figure 3.1: Discretization of the interval $[a,b]$ into $n+1$ subintervals and nodes $x_0,\ldots,x_{n+1}$.

A finite difference method is used to approximate the second derivative using a central-difference $\mathcal{O}((\Delta x)^2)$ formula [21, sec. 7.2]

$$w_i'' \approx \frac{w_{i-1} - 2w_i + w_{i+1}}{2\Delta x}. \tag{3.3}$$

Using the equation $-\mathbf{u}'' = \mathbf{f}$, the following formula is obtained

$$-\frac{w_{j-1} - 2w_j + w_{j+1}}{(\Delta x)^2} = f_j. \tag{3.4}$$

3.1. Case i: Homogeneous Dirichlet Boundary Conditions

Using $\mathbf{u} = 0$ on $\partial\Omega$, the values of $u(a) = w_0$ and $u(b) = w_{n+1}$ are both known to be zero. This implies that (3.4) has only to be used for $1 \le j \le n$.

For $j = 1$, (3.4) implies

$$-\frac{w_0 - 2w_1 + w_2}{(\Delta x)^2} = f_1. \tag{3.5}$$

Using the boundary condition $w_0 = 0$, (3.4) simplifies to

$$-\frac{-2w_1 + w_2}{(\Delta x)^2} = f_1. \tag{3.6}$$

Similarly, using $w_{n+1} = 0$ for $j = n$, (3.4) simplifies to

$$-\frac{w_{n-1} - 2w_n}{(\Delta x)^2} = f_n. \tag{3.7}$$

Putting (3.4), (3.6), and (3.7) together, gives

$$f_j = \begin{cases} -\dfrac{-2w_1 + w_2}{(\Delta x)^2} & j = 1 \\ -\dfrac{w_{j-1} - 2w_j + w_{j+1}}{(\Delta x)^2} & 2 \le j \le n-1 \\ -\dfrac{w_{n-1} - 2w_n}{(\Delta x)^2} & j = n \end{cases}, \tag{3.8}$$

or in matrix form, this is

$$\begin{aligned} &A\mathbf{w} = \mathbf{f}, \quad \text{with} \\ &A = \frac{1}{(\Delta x)^2} \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{n \times n}, \\ &\mathbf{w} = \begin{bmatrix} w_1 & w_2 & \dots & w_n \end{bmatrix}^T, \text{ and} \\ &\mathbf{f} = \begin{bmatrix} f_1 & f_2 & \dots & f_n \end{bmatrix}^T. \end{aligned} \tag{3.9}$$

This matrix is symmetric.

Proposition 3.1. *We will show that matrix A is positive definite. From this fact, it follows that A is non-singular and that* (3.9) *has a unique solution for every* $\mathbf{f}$.

Proof. We follow the proof from Slide 29 of [4].

We show that the matrix $T = (\Delta x)^2 A$ is positive definite. Let $\mathbf{z} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$.
Then,

$$\begin{aligned} \mathbf{z}^T T \mathbf{z} &= \sum_{i=1}^{n} 2z_i^2 - \sum_{i=1}^{n-1} z_i z_{i+1} - \sum_{i=1}^{n-1} z_{i+1} z_i, \\ &= \sum_{i=1}^{n} 2z_i^2 - \sum_{i=1}^{n-1} 2z_i z_{i+1}, \\ &= z_1^2 + (z_1^2 - 2z_1 z_2 + z_2^2) + (z_2^2 - 2z_2 z_3 + z_3^2) \\ &+ \cdots + (z_{n-1}^2 - 2z_{n-1} z_n + z_n^2) + z_n^2, \\ &= z_1^2 + (z_1 - z_2)^2 + (z_2 - z_3)^2 + \cdots + (z_{n-1} - z_n)^2 + z_n^2, \\ &= z_1^2 + \sum_{i=1}^{n-1} (z_i - z_{i+1})^2 + z_n^2 \\ &> 0, \end{aligned} \tag{3.10}$$

since the sum consists of non-negative terms, and at least one must be strictly positive when $\mathbf{z} \neq \mathbf{0}$.
Thus A is positive definite. □

3.2. Case ii: Homogeneous Neumann Boundary Conditions

Assuming homogeneous Neumann boundary conditions, i.e.,

$$\partial_n \mathbf{u} = 0 \quad \text{on } \partial\Omega, \tag{3.11}$$

this implies that the derivatives at the boundaries vanish: $w_0' = w_{n+1}' = 0$.

In this case, all values $w_0, w_1, \ldots, w_n, w_{n+1}$ are treated as unknowns.

Two different approaches to incorporate these boundary conditions are considered, each leading to a system matrix with distinct structural properties [11, ch. 3].

3.2.1. Default version

The standard procedure to use a central difference for the first derivative for this, two (virtual) points outside the domain are needed. Therefore the $x_{-1} = a - \Delta x$ and $x_{n+2} = b + \Delta x$ are considered.

A central difference scheme for the first derivate for w_0' is given by

$$w_0' = \frac{w_1 - w_{-1}}{2\Delta x}. \tag{3.12}$$

Using $w_0' = 0$, we obtain

$$w_{-1} = w_1. \tag{3.13}$$

Similarly, applying $w_{n+1}' = 0$ with the central difference approximation

$$w_{n+1}' = \frac{w_{n+2} - w_n}{2\Delta x}, \tag{3.14}$$

leads to the result

$$w_{n+2} = w_n. \tag{3.15}$$

Since w_0 and w_{n+1} are unknown, (3.4) is also applied at these points. Consequently, the boundary condition cannot be enforced at w_1 and w_n, and the central difference approximation (3.4) must also be used for these values.

For w_0, using (3.4) and (3.13) gives

$$-\frac{w_{-1} - 2w_0 + w_1}{(\Delta x)^2} = -\frac{-2w_0 + 2w_1}{(\Delta x)^2} = f_0. \tag{3.16}$$

In the same way, for $j = n+1$, combining (3.4) and (3.15) gives the following scheme

$$-\frac{2w_n - 2w_{n+1}}{(\Delta x)^2} = f_{n+1}. \tag{3.17}$$

Combining (3.4), (3.16) and (3.17) together gives

$$f_j = \begin{cases} -\dfrac{-2w_0 + 2w_1}{(\Delta x)^2} & j = 0 \\ -\dfrac{w_{j-1} - 2w_j + w_{j+1}}{(\Delta x)^2} & 1 \leq j \leq n\,, \\ -\dfrac{2w_n - 2w_{n+1}}{(\Delta x)^2} & j = n+1 \end{cases} \tag{3.18}$$

which is written in matrix notation as

$$
\begin{aligned}
&A\mathbf{w} = \mathbf{f}, \quad \text{with} \\
&A = \frac{1}{(\Delta x)^2}\begin{bmatrix} 2 & -2 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -2 & 2 \end{bmatrix} \in \mathbb{R}^{(n+2)\times(n+2)}, \\
&\mathbf{w} = \begin{bmatrix} w_0 & w_1 & \dots & w_{n+1} \end{bmatrix}^T, \text{ and} \\
&\mathbf{f} = \begin{bmatrix} f_0 & f_1 & \dots & f_{n+1} \end{bmatrix}^T.
\end{aligned} \tag{3.19}
$$

It is readily seen that this matrix is non-symmetric. Moreover, since all rows sum to zero, Theorem 2.1 implies that A is singular.

By the diagonal dominance of the matrix, it can be shown that the eigenvalues of A are in the non-negative part of the complex plane.

3.2.2. Symmetric version

Instead of applying a central difference scheme for the first derivative, a forward difference is used for w_0', and a backward difference is used for w_{n+1}'.

As in the non-symmetric formulation, all values $w_0, w_1, \dots, w_n, w_{n+1}$ are considered unknown.

The forward difference approximation for w_0' gives

$$w_0' = \frac{w_1 - w_0}{\Delta x} = 0, \tag{3.20}$$

from which it follows that

$$w_0 = w_1. \tag{3.21}$$

Thus (3.4) for $j = 1$ is given by

$$-\frac{w_1 - 2w_1 + w_2}{(\Delta x)^2} = f_1, \tag{3.22}$$

which simplifies to

$$-\frac{-w_1 + w_2}{(\Delta x)^2} = f_1. \tag{3.23}$$

Similarly, using $w_{n+1}' = 0$, and a backward difference scheme

$$w_{n+1} = w_n \tag{3.24}$$

is obtained, from which the finite difference scheme for $j = n$ is

$$-\frac{-w_{n-1} + w_n}{(\Delta x)^2} = f_n. \tag{3.25}$$

Combining (3.4), (3.23) and (3.25) gives

$$f_j = \begin{cases} -\dfrac{-w_1 + w_2}{(\Delta x)^2} & j = 1 \\ -\dfrac{w_{j-1} - 2w_j + w_{j+1}}{(\Delta x)^2} & 2 \le j \le n-1 \\ -\dfrac{w_{n-1} - w_n}{(\Delta x)^2} & j = n \end{cases}, \tag{3.26}$$

or in matrix form is this

$$
\begin{aligned}
&A\mathbf{w} = \mathbf{f}, \quad \text{with} \\
&A = \frac{1}{(\Delta x)^2}\begin{bmatrix} 1 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 1 \end{bmatrix} \in \mathbb{R}^{n\times n}, \\
&\mathbf{w} = \begin{bmatrix} w_1 & w_2 & \dots & w_n \end{bmatrix}^T, \text{ and} \\
&\mathbf{f} = \begin{bmatrix} f_1 & f_2 & \dots & f_n \end{bmatrix}^T.
\end{aligned} \tag{3.27}
$$

Unlike the matrix obtained using central differences in (3.19), the matrix in (3.27) is symmetric. However, since the elements of each row sum to zero, the matrix A remains singular.

By applying the Gershgorin circle theorem (Theorem 2.2), it can be shown that all eigenvalues of A are non-negative, due to its weakly diagonal dominance and symmetry.

3.3. Case iii: Mixed Boundary Conditions

In this case, there is a homogeneous Dirichlet boundary condition on $x = a$ and a homogeneous Neumann boundary condition on $x = b$. In other words $u(a) = u'(b) = 0$.

Therfore, the value for w_0 is known, and the value for w_{n+1} has to be determined.

3.3.1. Default version

For $j = 1$, the equation as given in (3.6) is used. For $j = n+1$, difference scheme (3.17) is chosen as it arises from approximating w'_{n+1} by central differences.

Combining these results and using (3.4) for $2 \le j \le n$, the following formula is obtained:

$$
f_j = \begin{cases} -\dfrac{-2w_1 + w_2}{(\Delta x)^2} & j = 1 \\ -\dfrac{w_{j-1} - 2w_j + w_{j+1}}{(\Delta x)^2} & 2 \le j \le n \\ -\dfrac{2w_n - 2w_{n-1}}{(\Delta x)^2} & j = n+1 \end{cases}. \tag{3.28}
$$

In matrix form is this

$$
\begin{aligned}
&A\mathbf{w} = \mathbf{f}, \quad \text{with} \\
&A = \frac{1}{(\Delta x)^2}\begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -2 & 2 \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}, \\
&\mathbf{w} = \begin{bmatrix} w_1 & w_2 & \dots & w_{n+1} \end{bmatrix}^T, \text{ and} \\
&\mathbf{f} = \begin{bmatrix} f_1 & f_2 & \dots & f_{n+1} \end{bmatrix}^T.
\end{aligned} \tag{3.29}
$$

Note that this matrix is non-symmetric.

Proposition 3.2. *We show that the matrix A from* (3.29) *is non-singular.*

Proof. We will prove the non-singularity by showing that the zero-vector is the only vector in the null space of A.

Let $\mathbf{v} \in N(A)$. Then $A\mathbf{v} = \mathbf{0}$. Applying this condition on the first row of the matrix gives

$$2v_1 - v_2 = 0 \quad \Longrightarrow \quad v_2 = 2v_1. \tag{3.30}$$

The second row gives

$$-v_1 + 2v_2 - v_3 = 0 \quad \Longrightarrow \quad 3v_1 - v_3 = 0 \Longrightarrow v_3 = 3v_1. \tag{3.31}$$

Let us show using induction that

$$v_k = kv_1 \quad \forall 1 \le k \le n. \tag{3.32}$$

Assume that (3.32) holds for all $i < k$. Using the $k-1$-th row of A and the induction hypothesis, we derive

$$-v_{k-2} + 2v_{k-1} - v_k = 0 \quad \Longrightarrow \quad -(k-2)v_1 + 2(k-1)v_1 - v_k = 0 \quad \Longrightarrow \quad v_k = kv_1. \tag{3.33}$$

Hence (3.32) holds for all k.

From the last row of the matrix, we have

$$-2v_{n-1} + 2v_n = 0 \quad \Longrightarrow \quad v_{n-1} = v_n. \tag{3.34}$$

Combining (3.32) and (3.34) gives:

$$nv_1 = v_n = v_{n-1} = (n-1)v_1 \quad \Longrightarrow \quad v_1 = 0. \tag{3.35}$$

Using (3.32) and (3.35), we conclude that $v_j = 0$ for all $1 \le j \le n$, and hence that $\mathbf{v} = \mathbf{0}$.

We have shown that the only vector in the kernel of A is the zero vector, which implies that A is non-singular. □

Proposition 3.3. *Furthermore, A has only eigenvalues in the positive real part of the complex plane.*

Proof. Applying Theorem 2.2, for each row i, the eigenvalues z satisfy

$$|z - a_{ii}| \le R_i, \tag{3.36}$$

where

$$R_i = \sum_{\substack{j=1 \\ j \ne i}}^{n+1} |a_{ij}|. \tag{3.37}$$

We compute the radii R_i and centers a_{ii} as follows:

- For $i = 1$:
$$a_{11} = \frac{2}{(\Delta x)^2}, \quad R_1 = \left| -\frac{1}{(\Delta x)^2} \right| = \frac{1}{(\Delta x)^2}. \tag{3.38}$$
- For $2 \le i \le n$:
$$a_{ii} = \frac{2}{(\Delta x)^2}, \quad R_i = \left| -\frac{1}{(\Delta x)^2} \right| + \left| -\frac{1}{(\Delta x)^2} \right| = \frac{2}{(\Delta x)^2}. \tag{3.39}$$
- For $i = n+1$:
$$a_{n+1,n+1} = \frac{2}{(\Delta x)^2}, \quad R_{n+1} = \left| -\frac{2}{(\Delta x)^2} \right| = \frac{2}{(\Delta x)^2}. \tag{3.40}$$

Therefore, all eigenvalues lie in the union of disks centered at $\frac{2}{(\Delta x)^2}$ with radii at most $\frac{2}{(\Delta x)^2}$:

$$\left| z - \frac{2}{(\Delta x)^2} \right| \leq \frac{2}{(\Delta x)^2}.$$

Since the leftmost point of these disks is at

$$\frac{2}{(\Delta x)^2} - \frac{2}{(\Delta x)^2} = 0, \tag{3.41}$$

the disks lie entirely in the closed right half of the complex plane (including zero).

However, since the matrix A is non-singular, it follows that

$$\lambda_i \neq 0 \quad \text{for all } \lambda_i, 1 \leq i \leq n+1. \tag{3.42}$$

Therefore, all eigenvalues of A satisfy

$$\text{Re}(\lambda_i) > 0. \tag{3.43}$$

□

3.3.2. Symmetric version

A symmetric formulation of the finite difference scheme with one Dirichlet and one Neumann boundary condition can also be derived, similar to the case with two Neumann conditions.

In Section 3.3.1, the Neumann condition at w_{n+1} was satisfied using central differences. Here, we instead apply a backward difference at the Neumann boundary, as in (3.25), while keeping the same expressions as before for the interior points and the Dirichlet condition at w_1 (i.e. (3.4) and (3.6), respectively). This leads to the following finite difference system:

$$f_j = \begin{cases} -\dfrac{-2w_1 + w_2}{(\Delta x)^2} & j = 1 \\ -\dfrac{w_{j-1} - 2w_j + w_{j+1}}{(\Delta x)^2} & 2 \leq j \leq n-1 \\ -\dfrac{w_{n-1} - w_n}{(\Delta x)^2} & j = n \end{cases}. \tag{3.44}$$

In matrix form, this is

$$\begin{aligned} &A\mathbf{w} = \mathbf{f}, \quad \text{with} \\ &A = \frac{1}{(\Delta x)^2} \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 1 \end{bmatrix} \in \mathbb{R}^{n \times n}, \\ &\mathbf{w} = \begin{bmatrix} w_1 & w_2 \dots & w_n \end{bmatrix}^T, \text{ and} \\ &\mathbf{f} = \begin{bmatrix} f_1 & f_2 & \dots & f_n \end{bmatrix}^T. \end{aligned} \tag{3.45}$$

This formulation yields a symmetric and non-singular system matrix A. The symmetry follows directly from the structure of the matrix, and the non-singularity can be proven using the same argument as for the non-symmetric case.

Proposition 3.4. *The matrix A from* (3.45) *is also positive definite.*

Proof. We follow the proof from Slide 29 of [4].

Let $\mathbf{z} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$, and define $T = (\Delta x)^2 A$. Then,

$$\begin{aligned}
\mathbf{z}^T T\mathbf{z} &= \sum_{i=1}^{n-1} 2z_i^2 + z_n^2 - \sum_{i=1}^{n-1} z_i z_{i+1} - \sum_{i=1}^{n-1} z_{i+1} z_i, \\
&= \sum_{i=1}^{n-1} 2z_i^2 + z_n^2 - \sum_{i=1}^{n-1} 2z_i z_{i+1}, \\
&= z_1^2 + (z_1^2 - 2z_1 z_2 + z_2^2) + (z_2^2 - 2z_2 z_3 + z_3^2) \\
&+ \cdots + (z_{n-1}^2 - 2z_{n-1} z_n + z_n^2), \\
&= z_1^2 + (z_1 - z_2)^2 + (z_2 - z_3)^2 + \cdots + (z_{n-1} - z_n)^2, \\
&= z_1^2 + \sum_{i=1}^{n-1} (z_i - z_{i+1})^2, \\
&> 0,
\end{aligned} \tag{3.46}$$

since all terms are non-negative, and $\mathbf{z} \neq \mathbf{0}$ implies that at least one term is strictly positive.

Hence, A is positive definite. □

4

Direct Methods

In this chapter, two direct methods: LU decomposition and Cholesky decomposition to solve a linear system of equations $A\mathbf{w} = \mathbf{f}$ are discussed.

4.1. General Idea

Solving a linear system of equations of the form $A\mathbf{w} = \mathbf{f}$ can be done using Gaussian elimination, but this approach typically involves a large number of arithmetic operations, especially for large systems. Therefore, more structured direct methods have been developed to improve computational efficiency.

The general idea of direct methods is to transform the original system into an equivalent one that is easier to solve. This is often achieved through matrix factorizations, where the matrix A is decomposed into the product of matrices with special structure. Once such a factorization is available, the system can be solved using for example forward and backward substitution, which are significantly computationally cheaper than solving the original system directly [8, sec. 5.1].

Direct methods are deterministic and aim to provide an accurate solution in a finite number of steps, assuming exact arithmetic.

4.2. Methods

4.2.1. LU Decomposition

The goal of LU decomposition is to express a matrix $A \in \mathbb{R}^{n\times n}$ as the product of two easily invertible matrices: a lower triangular matrix L and an upper triangular matrix U, such that $A = LU$. This factorization allows for efficient solution of linear systems, as triangular systems can be solved in $\mathcal{O}(n^2)$ operations using forward and backward substitution.

The matrix U is obtained by applying Gaussian elimination to reduce A to row-echelon form, while the matrix L records the operations performed during this process. This procedure is outlined in Algorithm 1 [6, sec. 10.2].

Once the LU decomposition is obtained, the original system $A\mathbf{w} = \mathbf{f}$ can be solved in two steps:

$$\begin{aligned} A\mathbf{w} = \mathbf{f} \quad &\Longleftrightarrow \\ LU\mathbf{w} = \mathbf{f} \quad &\Longleftrightarrow \\ U\mathbf{w} = \underbrace{L^{-1}\mathbf{f}}_{\mathbf{y}} \quad &\Longleftrightarrow \\ \mathbf{w} = U^{-1}\mathbf{y}. \end{aligned} \tag{4.1}$$

In practice, the inverses L^{-1} and U^{-1} are not computed explicitly. Instead, we solve the two triangular systems:

$$L\mathbf{y} = \mathbf{f} \quad \text{(forward substitution)}, \qquad U\mathbf{w} = \mathbf{y} \quad \text{(backward substitution)}.$$

Algorithm 1 LU Decomposition

Initialize L as the $n \times n$ identity matrix.
Initialize $U \leftarrow A$
for $i = 1$ to n **do**
    pivot $\leftarrow U[i][i]$
    for $j = i+1$ to n **do**
        $c \leftarrow \dfrac{U[j][i]}{\text{pivot}}$
        $L[j][i] \leftarrow c$
        for $k = 1$ to n **do**
            $U[j][k] \leftarrow U[j][k] - c \cdot U[i][k]$
        end for
    end for
    $L[i][i] \leftarrow 1$
end for
return L, U

This process is shown in Algorithm 2.

Algorithm 2 Solve $A\mathbf{w} = \mathbf{f}$ using LU decomposition

Require: Lower triangular matrix L, upper triangular matrix U, right-hand side vector $\mathbf{f}$
Forward substitution: solve $L\mathbf{y} = \mathbf{f}$
for $i = 1$ to n **do**
    $c[i] \leftarrow f[i] - \sum_{j=1}^{i-1} L[i][j] \cdot c[j]$
end for
Back substitution: solve $U\mathbf{w} = \mathbf{c}$
for $i = n$ down to 1 **do**
    $w[i] \leftarrow \dfrac{1}{U[i][i]} \left(y[i] - \sum_{j=i+1}^{n} U[i][j] \cdot w[j] \right)$
end for
return w

For a solution to exist, both L and U must be invertible. Since L is constructed with ones on its diagonal, it is always invertible. The invertibility of U depends on the matrix A. In particular, U will contain a zero on the diagonal if A is singular, making the decomposition invalid for solving the system.

In general, LU decomposition without any modifications may fail or become numerically unstable if A has small or zero pivot elements. To improve numerical stability and ensure the decomposition is always possible when A is nonsingular, pivoting (typically partial pivoting) is introduced. LU decomposition with pivoting produces a factorization of the form $PA = LU$, where P is a permutation matrix. The algorithms for LU decomposition with pivoting are included in Appendix A.

However, in the special case of structured matrices such as tridiagonal matrices, pivoting is often unnecessary. For well-conditioned tridiagonal systems, LU decomposition without pivoting is both sufficient and computationally efficient, with complexity reduced from $\mathcal{O}(n^3)$ to $\mathcal{O}(n^2)$ [6, sec. 10.2].

4.2.2. Cholesky Decomposition

Cholesky decomposition is a direct method for solving linear systems when the coefficient matrix $A \in \mathbb{R}^{n\times n}$ is symmetric and positive definite. In this case, A can be uniquely factored as

$$A = LL^T, \tag{4.2}$$

where L is a lower triangular matrix with positive diagonal entries, and L^T is its transpose.

This factorization offers both computational efficiency and numerical stability. Compared to LU decomposition, Cholesky decomposition requires roughly half the number of operations, making it especially advantageous for large systems where the symmetric positive definite condition holds.

The matrix L is constructed entry-by-entry. Similar to Gaussian elimination, the algorithm incrementally eliminates the influence of previously computed entries. For each entry L_{ij}, the corresponding value of A_{ij} is corrected by subtracting the dot product of the i-th and j-th rows (up to index $j-1$). The diagonal elements ($i = j$) require taking a square root after subtraction, while off-diagonal entries are divided by the appropriate diagonal element L_{jj}. The complete procedure is outlined in Algorithm 3 [9].

Algorithm 3 Cholesky Decomposition

for $i = 1$ to n **do**
  for $j = 1$ to i **do**
    sum $\leftarrow A[i][j]$
    for $k = 1$ to $j-1$ **do**
      sum $\leftarrow$ sum $- L[i][k] \cdot L[j][k]$
    end for
    if $i = j$ **then**
      $L[i][j] \leftarrow \sqrt{\text{sum}}$
    else
      $L[i][j] \leftarrow \dfrac{\text{sum}}{L[j][j]}$
    end if
  end for
end for
return L

Once the Cholesky factor L is known, the system $A\mathbf{w} = \mathbf{f}$ can be solved efficiently by exploiting the identity $A = LL^T$. The solution process is structured as:

$$\begin{aligned} A\mathbf{w} = \mathbf{f} \quad &\Longleftrightarrow \\ LL^T\mathbf{w} = \mathbf{f} \quad &\Longleftrightarrow \\ L^T\mathbf{w} = \underbrace{L^{-1}\mathbf{f}}_{\mathbf{y}} \quad &\Longleftrightarrow \\ \mathbf{w} = \left(L^T\right)^{-1}\mathbf{y} \end{aligned} \tag{4.3}$$

As with LU decomposition, explicit matrix inversion is avoided. Instead, the solution proceeds in two stages using substitution methods:

1. Forward substitution: Solve $L\mathbf{y} = \mathbf{f}$.
2. Backward substitution: Solve $L^T\mathbf{w} = \mathbf{y}$.

The complete algorithm for solving the system using the Cholesky decomposition is presented in Algorithm 4 [9].

Unlike general LU decomposition, Cholesky decomposition does not require pivoting, since the positive definiteness of the matrix guarantees that all pivots are strictly positive [7, thm. 4.2.7]. Additionally, it is more storage-efficient, as only a single lower triangular matrix needs to be stored.

Algorithm 4 Solve $A\mathbf{w} = \mathbf{f}$ using Cholesky decomposition

Require: Lower triangular matrix L, right-hand side vector $\mathbf{f}$

Forward substitution: solve $L\mathbf{y} = \mathbf{f}$
for $i = 1$ to n **do**
    $y[i] \leftarrow \frac{1}{L[i][i]} \left(f[i] - \sum_{j=1}^{i-1} L[i][j] \cdot y[j] \right)$
end for
Backward substitution: solve $L^T\mathbf{w} = \mathbf{y}$
for $i = n$ down to 1 **do**
    $w[i] \leftarrow \frac{1}{L[i][i]} \left(y[i] - \sum_{j=i+1}^{n} L[j][i] \cdot w[j] \right)$
end for
return w

4.3. Comparison

To assess the effectiveness of LU and Cholesky decompositions under different boundary conditions, Table 4.1 summarizes whether each method succeeds or fails for several representative cases. The outcome depends heavily on matrix properties such as symmetry, invertibility, and positive definiteness.

Case	LU Decomposition Result	Cholesky Decomposition Result
i: Dirichlet boundary conditions	LU decomposition gives a correct outcome. The resulting matrix is invertible and positive definite.	Cholesky decomposition also gives a correct result, as the matrix is symmetric and positive definite.
ii[a]: Neumann boundary conditions	LU decomposition fails. The matrix is singular, resulting in a zero pivot in U, leading to division by zero during back substitution.	Cholesky decomposition also fails because the matrix is neither symmetric nor positive definite.
ii[b]: Symmetric version of ii[a]	LU decomposition fails due to the matrix being singular.	Cholesky decomposition fails as the matrix remains singular and thus not positive definite.
iii[a]: Mixed boundary conditions	LU decomposition succeeds. The matrix is non-singular, and both L and U have non-zero diagonal entries.	Cholesky decomposition fails, as the matrix is not positive definite despite being solvable via LU.
iii[b]: Symmetric version of iii[a]	LU decomposition gives a correct result, as invertibility is preserved.	Cholesky decomposition also succeeds, since the matrix is symmetric and positive definite.

Table 4.1: Comparison of LU and Cholesky decomposition outcomes under different boundary conditions.

From Table 4.1, we conclude the following:

- LU decomposition can solve systems as long as the matrix is invertible, even if the matrix is not symmetric or positive definite.
- Cholesky decomposition is more restrictive: it requires the matrix to be symmetric and positive definite. This condition is not always satisfied, especially under Neumann or non-symmetrized mixed boundary conditions.
- In practice, Cholesky is computationally more efficient and numerically stable when applicable, but LU offers broader applicability.

5

Iterative Methods

In this section, several iterative methods for solving a linear system of equations $A\mathbf{w} = \mathbf{f}$ are discussed.

5.1. General Idea

The idea behind an iterative method is to obtain approximations $\mathbf{w}^{(k)}$ (iterate) of the solution, starting from an initial guess $\mathbf{w}^{(0)}$, such that one generates a sequence $\{\mathbf{w}^{(k)}\}$ with $\mathbf{w}^{(k)} \to \mathbf{w}$ as $k \to \infty$, where $\mathbf{w}$ is the exact solution of the system.

The distance between each iterate and the exact solution is called the error, defined as

$$\mathbf{e}^{(k)} = \mathbf{w} - \mathbf{w}^{(k)}, \tag{5.1}$$

and we want $\mathbf{e}^{(k)} \to \mathbf{0}$ as $k \to \infty$.

In most cases, the error $\mathbf{e}^{(k)}$ cannot be computed directly, since it requires knowledge of the exact solution $\mathbf{w}$, which would make the iteration process unnecessary. Instead, the residual vector at iteration k is defined as

$$\mathbf{r}^{(k)} = \mathbf{f} - A\mathbf{w}^{(k)}, \tag{5.2}$$

which is always computable. Note that since $A\mathbf{w} = \mathbf{f}$, we can rewrite (5.2) as

$$\sigma^{(k)} = A(\mathbf{w} - \mathbf{w}^{(k)}) = A\mathbf{e}^{(k)}. \tag{5.3}$$

The residual vector serves as a practical and accessible indicator of convergence in iterative methods, as it provides a measure of how accurately the approximate solution satisfies the linear system. Nonetheless, it is important to recognize that a small residual does not necessarily imply a small error in the solution. In particular, for ill-conditioned matrices, the residual may significantly underestimate the true error. Consequently, while the residual is a useful tool, its interpretation should be made with care, and, where possible, complemented with additional estimates or bounds relating the residual to the actual error.

In an iterative method, the matrix A is decomposed into two parts M and N, such that $A = M + N$. The matrix M is chosen to be a non-singular matrix and such that solving systems of the form $M\mathbf{y} = \mathbf{c}$ is straightforward.

Solving $A\mathbf{w} = \mathbf{f}$ then gives rise to an iterating scheme as follows

$$\begin{aligned} A\mathbf{w} = \mathbf{f} \quad &\iff \\ M\mathbf{w} + N\mathbf{w} = \mathbf{f} \quad &\iff \\ M\mathbf{w} = \mathbf{f} - N\mathbf{w}. \end{aligned} \tag{5.4}$$

In this formulation, the left-hand side uses the new iterate $\mathbf{w}^{(k+1)}$, while the right-hand side uses the previous iterate $\mathbf{w}^{(k)}$:

$$\begin{aligned} M\mathbf{w}^{(k+1)} &= \mathbf{f} - N\mathbf{w}^{(k)} \qquad \Longleftrightarrow \\ \mathbf{w}^{(k+1)} &= M^{-1}(\mathbf{f} - N\mathbf{w}^{(k)}) \\ &= M^{-1}\mathbf{f} - M^{-1}(A-M)\mathbf{w}^{(k)} \\ &= \mathbf{w}^{(k)} + M^{-1}(\mathbf{f} - A\mathbf{w}^{(k)}) \\ &= \mathbf{w}^{(k)} + M^{-1}\mathbf{r}^{(k)}. \end{aligned} \tag{5.5}$$

From this, we see that

$$\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)} = M^{-1}\mathbf{r}^{(k)}. \tag{5.6}$$

From iterative formula (5.5) and using (5.3), the recursion for the error vector becomes:

$$\begin{aligned} \mathbf{e}^{(k+1)} &\underset{(5.5)}{=} \mathbf{w} - \mathbf{w}^{(k+1)} \\ &\underset{(5.3)}{=} \mathbf{w} - \mathbf{w}^{(k)} - M^{-1}\mathbf{r}^{(k)} \\ &= \mathbf{e}^{(k)} - M^{-1}A\mathbf{e}^{(k)} \\ &= (I - M^{-1}A)\mathbf{e}^{(k)}, \end{aligned} \tag{5.7}$$

and for the residual vector, iterative formula (5.5) gives

$$\begin{aligned} \mathbf{r}^{(k+1)} &\underset{(5.5)}{=} \mathbf{f} - A\mathbf{w}^{(k+1)} \\ &= \mathbf{f} - A\mathbf{w}^{(k)} - AM^{-1}\mathbf{r}^{(k)} \\ &= (I - AM^{-1})\mathbf{r}^{(k)}. \end{aligned} \tag{5.8}$$

Note that the matrices $I - M^{-1}A$ and $I - AM^{-1}$ are related in the following way:

$$\begin{aligned} I - M^{-1}A &= A^{-1}A - A^{-1}AM^{-1}A \\ &= A^{-1}\left(I - AM^{-1}\right)A. \end{aligned} \tag{5.9}$$

This implies that the matrices in the error and residual recursions have the same spectrum (Proposition 2.1).

Based on this formula, we define the iteration matrix as

$$B = I - M^{-1}A = I - M^{-1}(M+N) = I - I - M^{-1}N = -M^{-1}N, \tag{5.10}$$

which can be used to define stopping criteria [20, sec. 5.2].

There are several criteria to determine whether an iterative method has converged. In this context, we consider the following two commonly used convergence criteria:

1. Relative change criterion:
   The iteration is considered to have converged if the relative change between two successive iterates is sufficiently small:
   $$\frac{\left\|\mathbf{w}^{(k+1)} - \mathbf{w}^{(k)}\right\|}{\left\|\mathbf{w}^{(k+1)}\right\|} < \epsilon. \tag{5.11}$$
   This condition ensures that the iterates are no longer changing significantly, indicating stabilization of the solution [21, sec. 4.2].

2. Relative residual norm criterion:
   The iteration is also considered to have converged if the norm of the residual vector is small relative to the norm of the right-hand side:
   $$\frac{\left\|\mathbf{r}^{(k)}\right\|}{\|\mathbf{f}\|} < \epsilon. \tag{5.12}$$
   This condition measures how well the current iterate satisfies the original system of equations [12, sec. 2.3].

Note that the denominator of both criterium are connected by (5.6).

To achieve convergence in an iterative method, we require $\mathbf{e}^{(k)} \to \mathbf{0}$ as $k \to \infty$. Using the recurrence relation for the error vector from (5.7), we obtain:

$$\begin{aligned} \mathbf{e}^{(k)} &= (I - \underset{B}{M^{-1}A})\mathbf{e}^{(k-1)} \\ &= B^2\mathbf{e}^{(k-2)} \\ &= \dots \\ &= B^k\mathbf{e}^{(0)}. \end{aligned} \tag{5.13}$$

Proposition 5.1. *A necessary and sufficient condition for convergence of the iterative method is that the spectral radius of the iteration matrix satisfies*

$$\rho(B) = \rho(I - M^{-1}A) < 1 \quad \iff \quad \{\mathbf{w}^{(k)}\}_{k=0}^{\infty} \text{ converges to } \mathbf{w}. \tag{5.14}$$

Proof. We follow the proof of Section 7.10 of [14].

Let B be an iteration matrix of an iterative method. Assume $B^k \to 0$ when $k \to \infty$. We will show $\rho(B) < 1$.

Let λ be an eigenvalue of B and let $\mathbf{v}$ be an eigenvector corresponding to λ. We have $B\mathbf{v} = \lambda v$, which implies $B^k\mathbf{v} = \lambda^k\mathbf{v}$. Then:

$$\begin{aligned} 0 &= \left(\lim_{k\to\infty} B^k\right)\mathbf{v} \\ &= \lim_{k\to\infty} (B^k\mathbf{v}) \\ &= \lim_{k\to\infty} \lambda^k\mathbf{v} \\ &= \mathbf{v} \lim_{k\to\infty} \lambda^k. \end{aligned}$$

Since $\mathbf{v} \neq \mathbf{0}$, we have $\lambda^k \to 0$, which implies $|\lambda| < 1$. As λ was arbitrary, in particular $\rho(B) < 1$.

Conversely, assume $\rho(B) < 1$. Let $P, J \in \mathbb{C}^{n\times n}$ such that $J = P^{-1}BP$, with P non-singular and J block diagonal, according to Theorem 2.3.

Then $B^k = PJ^kP^{-1}$, and

$$J^k = \begin{bmatrix} J_{m_1}^k(\lambda_1) & & & \\ & J_{m_2}^k(\lambda_2) & & \\ & & \ddots & \\ & & & J_{m_s}^k(\lambda_s) \end{bmatrix}, \tag{5.15}$$

with

$$J_{m_i}^k(\lambda_i) = \begin{bmatrix} \lambda_i^k & \binom{k}{1}\lambda_i^{k-1} & \binom{k}{2}\lambda_i^{k-2} & \dots & \binom{k}{m_i-1}\lambda_i^{k-m_i+1} \\ & \lambda_i^k & \binom{k}{1}\lambda_i^{k-1} & \dots & \binom{k}{m_i-2}\lambda_i^{k-m_i+2} \\ & & \ddots & \ddots & \vdots \\ & & & \lambda_i^k & \binom{k}{1}\lambda_i^{k-1} \\ & & & & \lambda_i^k \end{bmatrix} \in \mathbb{C}^{m_i\times m_i}. \tag{5.16}$$

Using the assumption $\rho(B) < 1$ gives $|\lambda_i| < 1$ for all i, thus

$$\lambda_i^k \to 0 \quad \text{as } k \to \infty, \tag{5.17}$$

and

$$\left|\binom{k}{j}\lambda_i^k\right| = \left|\frac{k(k-1)\dots(k-j+1)}{j!}\lambda_i^k\right| \leq \left|\frac{k^j}{j!}\lambda_i^k\right| \leq \frac{k^j}{j!}|\lambda_i|^{k-j} \to 0 \text{ as } k \to \infty. \tag{5.18}$$

This implies

$$\lim_{k\to\infty} J_{m_i}^k(\lambda_i) = \mathbf{0} \in \mathbb{R}^{n\times n}, \text{and hence } \lim_{k\to\infty} J^k = \mathbf{0} \in \mathbb{R}^{n\times n}. \tag{5.19}$$

Finally,

$$\lim_{k\to\infty} B^k = \lim_{k\to\infty} PJ^kP^{-1} = P\left(\lim_{k\to\infty} J^k\right)P^{-1} = \mathbf{0} \in \mathbb{R}^{n\times n}. \tag{5.20}$$

Which finishes the proof. □

This condition not only determines whether the iteration method converges but also provides insight into the rate of convergence. If M is a good approximation of A, then the spectral radius $\rho(I - M^{-1}A)$ is significantly less than 1, and we expect a fast convergence of the method. Conversely, if $\rho(I - M^{-1}A)$ is close to 1, the convergence is slow, indicating that M is a bad approximation of A [20, sec. 5.4.1].

5.2. Methods

5.2.1. Jacobi

In the Jacobi method, the matrix A is split into $A = D + (L + U)$, where:

- D is the diagonal part of A,
- L is the strictly lower triangular part, and
- U is the strictly upper triangular part.

See also Figure 5.1.

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} = D + (L + U)$$

Figure 5.1: Visualization of the Jacobi decomposition $A = D + (L + U)$, where D is the diagonal, L is the strictly lower triangular, and U the strictly upper triangular part of A.

Using this decomposition, the iteration matrices are defined as

$$M = D, \quad N = L + U. \tag{5.21}$$

This yields the iterative scheme

$$\mathbf{w}^{(k+1)} = D^{-1}\left(\mathbf{f} - (L+U)\mathbf{w}^{(k)}\right), \tag{5.22}$$

and the iteration matrix B_{Jac} is defined as

$$B_{\text{Jac}} = -D^{-1}(L+U). \tag{5.23}$$

Let $\mathbf{w}^{(k)}$ denote the approximation of the solution at iteration k, and let $w_i^{(k)}$ denote its ith component. Then, the Jacobi update rule for each component is given by

$$w_i^{(k+1)} = \frac{1}{a_{ii}}\left(f_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij}w_j^{(k)}\right), \tag{5.24}$$

for $i = 1, 2, \ldots, n$.

This formula computes the new value w_i^{k+1} using only values from the previous iterate $\mathbf{w}^{(k)}$, making the Jacobi method a fully decoupled iteration. This independence allows the Jacobi method to be parallelizable.

For the Jacobi method to be applicable, it is required that $a_{ii} \neq 0$ for all i, i.e. the diagonal entries of A must be non-zero so that the division in Equation (5.24) is defined.

Proposition 5.2. *The convergence of the Jacobi method is not guaranteed for all matrices A. However, it is guaranteed to converge if the matrix A is strictly diagonally dominant.*

Proof. We follow the proof of Theorem 2.1 of [19].

$$\begin{aligned} \rho(M^{-1}N) &\leq \left\|M^{-1}N\right\|_\infty = \left\|D^{-1}(L+U)\right\|_\infty \\ &= \max_{1\leq i\leq n} \sum_{j=1}^{n} (D^{-1}(L+U))_{ij} \\ &= \max_{1\leq i\leq n} \sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{|a_{ij}|}{|a_{ii}|} \\ &< \frac{|a_{ii}|}{|a_{ii}|} = 1. \end{aligned} \tag{5.25}$$

The spectral radius of the iteration matrix is thus strictly less than 1, which implies the method is convergent. Thus solving $A\mathbf{w} = \mathbf{f}$ give a solution $\mathbf{w}$, independent of the initial guess $\mathbf{w}^{(0)}$. □

The method is summarized in Algorithm 5 [17, sec. 4.1].

Algorithm 5 Solve $A\mathbf{w} = \mathbf{f}$ using Jacobi iteration

Require: Square matrix $A \in \mathbb{R}^{n\times n}$, right-hand side vector $\mathbf{f} \in \mathbb{R}^n$,
initial guess $\mathbf{w}^{(0)} \in \mathbb{R}^n$, maximum number of iterations $n_{\max}$,
convergence tolerance $\epsilon > 0$

$k \leftarrow 0$
while $k < n_{\max}$ **do**
  Initialize new iterate: $\mathbf{w}^{(k+1)} \leftarrow \mathbf{0}$
  for $i = 1$ to n **do**
    $w^{(k+1)}[i] \leftarrow \frac{1}{a[i][i]}\left(f[i] - \sum_{\substack{j=1 \\ j\neq i}}^{n} a[i][j] \cdot w^{(k)}[j]\right)$
  end for
  if converged **then**
    break
  end if
  $k \leftarrow k + 1$
end while
return $\mathbf{w}^{(k+1)}$

5.2.2. Gauss-Seidel

The Gauss-Seidel method is another iterative method for solving a linear system of equations. It improves upon the Jacobi method by taking into account the most recent updates for $\mathbf{w}$ during each iteration step.

The matrix A is split into two parts:

$$A = (L + D) + U. \tag{5.26}$$

This decomposition results into the following iteration formula:

$$\mathbf{w}^{(k+1)} = (L + D)^{-1}\left(\mathbf{f} - U\mathbf{w}^{(k)}\right), \tag{5.27}$$

with corresponding iteration matrix

$$B_{\text{GS}} = (L + D)^{-1}U. \tag{5.28}$$

Because $M = L + D$ is a lower triangular matrix, solving systems of the form $M\mathbf{y} = \mathbf{c}$ is straightforward using forward substitution (assuming non-zero diagonal entries).

In component-wise form, the Gauss-Seidel iteration updates each component $w_i^{(k+1)}$ as

$$w_i^{(k+1)} = \frac{1}{a_{ii}}\left(f_i - \sum_{j=1}^{i-1} a_{ij}w_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}w_j^{(k)}\right) \tag{5.29}$$

for $i = 1, 2, \ldots n$.

It can be observed that in this formula, the key difference compared to the Jacobi method, is that the updated values $w_j^{(k+1)}$ are used for indices $j < i$, instead of relying only of the values of the previous iteration $w_j^{(k)}$, as in the Jacobi method. While this change leads to faster convergence (show later), it also introduces a drawback: the iteration becomes sequential. Since each update depends on the latest computed values within the same iteration step, the method is much more difficult to parallelize efficiently.

The method is summarized in Algorithm 6 [12, sec. 1.4].

A variant of the Gauss-Seidel method uses the decomposition

$$A = (U + D) + L, \tag{5.30}$$

where $M = U + D$ and $N = L$.

This leads to the backward Gauss-Seidel iteration, where the updated values $w_j^{(k+1)}$ for indices $j > i$ are used within the same iteration step. The component-wise update formula becomes

$$w_i^{(k+1)} = \frac{1}{a_{ii}}\left(f_i - \sum_{j=1}^{i-1} a_{ij}w_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}w_j^{(k+1)}\right), \tag{5.31}$$

for $i = n, n-1, \ldots, 1$.

In this scheme, the iteration proceeds backward through the components, updating $w_n^{(k+1)}$ first and $w_1^{(k+1)}$ last. Similar to the forward Gauss-Seidel method, this approach also introduces sequential dependency, which affects parallelizability.

The next section shows that the convergence rate of Gauss-Seidel is twice that of Jacobi when A is tridiagonal (more generally, when A is consistently ordered).

Algorithm 6 Solve $A\mathbf{w} = \mathbf{f}$ using Gauss-Seidel iteration

Require: Square matrix $A \in \mathbb{R}^{n\times n}$, right-hand side vector $\mathbf{f} \in \mathbb{R}^n$,
initial guess $\mathbf{w}^{(0)} \in \mathbb{R}^n$, maximum number of iterations $n_{\max}$,
convergence tolerance $\epsilon > 0$

$k \leftarrow 0$
while $k < n_{\max}$ **do**
  Initialize new iterate: $\mathbf{w}^{(k+1)} \leftarrow \mathbf{0}$
  for $i = 1$ to n **do**
    $w^{(k+1)}[i] \leftarrow \frac{1}{a[i][i]}\left(f[i] - \sum_{j=1}^{i-1} a[i][j]\cdot w^{(k+1)}[j] - \sum_{j=i+1}^{n} a[i][j]\cdot w^{(k)}[j]\right)$
  end for
  if converged **then**
    break
  end if
  $k \leftarrow k+1$
end while
return $\mathbf{w}^{(k+1)}$

5.2.3. Successive Over-Relaxation

The Successive Over-Relaxation (SOR) method is an extension of the Gauss-Seidel method that introduces a relaxation parameter ω with the goal of accelerating convergence. As in the Gauss-Seidel method, each component of the solution vector $\mathbf{w}$ is updated sequentially using the most recent values. The key difference is that in SOR, each update is a weighted combination of the newly computed value and previous value, potentially leading to faster convergence. In this section, we use the lower-triangular variant of SOR.

Mathematically, the SOR update rule modifies the Gauss-Seidel formula as follows:

$$w_i^{(k+1)} = (1-\omega)\, w_i^{(k)} + \frac{\omega}{a_{ii}}\left(f_i - \sum_{j=1}^{i-1} a_{ij}\, w_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}\, w_j^{(k)}\right), \tag{5.32}$$

where ω is the relaxation parameter. Comparing this with the Gauss-Seidel update rule (Equation (5.29)), we observe that the SOR method applies a weight ω to the Gauss-Seidel update term, and combines it with the previous iterate $w_i^{(k)}$ scaled by $1-\omega$.

Proposition 5.3. $\rho(B_{SOR}) \geq |\omega - 1|$. *Therefore, a necessary condition for convergence is* $0 < \omega < 2$ *[5, thm. 6.4].*

Proof. We follow the proof of Theorem 6.4 of [5].

Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of the SOR iteration matrix B_{SOR}. Consider the characteristic polynomial:

$$\begin{aligned}\phi(\lambda) &= \det(\lambda I - B_{\text{SOR}}) \\ &= \det\big((I-\omega L)(\lambda I - B_{\text{SOR}})\big) \\ &= \det\big((\lambda+\omega-1)I - \omega\lambda L - \omega U\big),\end{aligned} \tag{5.33}$$

where L and U are the strict lower and upper triangular parts of the system matrix A, respectively.

Evaluating the characteristic polynomial at $\lambda = 0$, we obtain

$$\begin{aligned}\phi(0) &= \pm\prod_{i=1}^{n}\lambda_i \\ &= \pm\det((\omega-1)I) \\ &= \pm(\omega-1)^n.\end{aligned} \tag{5.34}$$

It follows that

$$\left|\prod_{i=1}^{n}\lambda_i\right| = |\omega-1|^n, \tag{5.35}$$

which implies that at least one eigenvalue λ_i must satisfy

$$|\lambda_i| \geq |\omega - 1|. \tag{5.36}$$

Therefore,

$$\rho(B_{\text{SOR}}) = \max_i |\lambda_i| \geq |\omega - 1|. \tag{5.37}$$

For the method to converge, we require $\rho(B_{\text{SOR}}) < 1$, which implies

$$|\omega - 1| < 1 \quad \Longleftrightarrow \quad \omega \in (0,2). \tag{5.38}$$

□

When $\omega = 1$, the method reduces to Gauss-Seidel iteration. For $0 < \omega < 1$, the method is said to be under-relaxed, which may improve stability in some cases. For $1 < \omega < 2$, the method is over-relaxed, and if ω is chosen well, convergence can be much faster.

The iteration matrix for SOR is given by

$$B_{\text{SOR}} = (D - \omega L)^{-1} ((1-\omega)D + \omega U). \tag{5.39}$$

The algorithm is detailedly described in Algorithm 7 [22].

Algorithm 7 Solve $A\mathbf{w} = \mathbf{f}$ using Successive Over-Relaxation

Require: Square matrix $A \in \mathbb{R}^{n\times n}$, right-hand side vector $\mathbf{f} \in \mathbb{R}^n$,
relaxation factor $\omega \in \mathbb{R}$, initial guess $\mathbf{w}^{(0)} \in \mathbb{R}^n$,
maximum number of iterations $n_{\max}$, convergence tolerance $\epsilon > 0$

$k \leftarrow 0$
while $k < n_{\max}$ **do**
  Initialize new iterate: $\mathbf{w}^{(k+1)} \leftarrow 0$
  for $i = 1$ to n **do**
    $c \leftarrow \frac{1}{a[i][i]} \left(f[i] - \sum_{j=1}^{i-1} a[i][j] \cdot w^{(k+1)}[j] - \sum_{j=i+1}^{n} a[i][j] \cdot w^{(k)}[j] \right)$
    $w^{(k+1)}[i] \leftarrow \omega \cdot c + (1-\omega) \cdot w^{(k)}[i]$
  end for
  if converged **then**
    break
  end if
  $k \leftarrow k + 1$
end while
return $\mathbf{w}^{(k+1)}$

The same convergence criteria as for the Jacobi and Gauss-Seidel methods, given by Equations (5.11) and (5.12), are also used for SOR.

To determine the optimal value of the relaxation parameter ω, we analyze the spectral radius of the iteration matrix.

Theorem 5.1. *Let B_{Jac} be the iteration matrix corresponding to the Jacobi method. For a matrix $A \in \mathbb{R}^{n\times n}$ that is consistently ordered, and assuming B_{Jac} has real eigenvalues with $\rho(B_{Jac}) < 1$, the optimal relaxation parameter ω_{opt} for the SOR method, according to [20, thm. 5.5.5], is given by:*

$$\omega_{opt} = \frac{2}{1 + \sqrt{1 - (\rho(B_{Jac}))^2}}. \tag{5.40}$$

Proof: See the proof of Theorem 5.5.5 of [20].

An example of applying this Theorem is for the matrix A from (3.9) with Dirichlet boundary conditions, we have:

$$D = 2I, \quad L+U = \text{matrix with } (-1) \text{ on the sub- and super-diagonals.} \tag{5.41}$$

which leads to the Jacobi iteration matrix

$$B_{\text{Jac}} = \frac{1}{2}(L+U). \tag{5.42}$$

According to [20, sec. 5.5], the spectral radius of this matrix satisfies

$$\rho(B_{\text{Jac}}) = 1 - \frac{\pi^2}{2}(\Delta x)^2 + \mathcal{O}((\Delta x)^4). \tag{5.43}$$

Substituting this expression into Equation (5.40) yields

$$\omega_{\text{opt}} = \frac{2}{1+\sqrt{1-\left(1-\frac{\pi^2}{2}(\Delta x)^2+\mathcal{O}((\Delta x)^4)\right)^2}} \approx \frac{2}{1+\mathcal{O}((\Delta x)^2)}. \tag{5.44}$$

As $\Delta x \to 0$, it can easily be seen that $\omega_{\text{opt}} \to 2$. Therefore, since the domain for ω is $(0,2)$, $\omega = 1.99$ is used.

Theorem 5.2. *For a consistently ordered matrix A and a relaxation parameter ω in $(0,2)$, the eigenvalues λ of the Jacobi iteration matrix B_{Jac} and μ of the SOR iteration matrix B_{SOR} are related by the identity [16, thm. 3.5]*

$$\sqrt{\mu}\omega\lambda = \mu + \omega - 1. \tag{5.45}$$

Proof. We follow the proof of Theorem 3.5 of [16].
Let λ and μ be eigenvalues of the Jacobi and SOR iteration matrix B_{Jac} and B_{SOR}, respectively. Suppose $\mathbf{v}$ is an eigenvector such that:

$$B_{\text{SOR}}\mathbf{v} = \mu\mathbf{v}. \tag{5.46}$$

Using definition (5.39) of the SOR iteration matrix, this can be rewritten as:

$$\left((1-\omega)I - \omega D^{-1}U\right)\mathbf{v} = \mu\left(I + \omega D^{-1}L\right)\mathbf{v}. \tag{5.47}$$

Rearranging terms gives:

$$(\mu+\omega-1)\mathbf{v} = -\sqrt{\mu}\omega\left(\sqrt{\mu}D^{-1}L + \frac{1}{\sqrt{\mu}}D^{-1}U\right) = \sqrt{\mu}\omega B(\sqrt{\mu})\mathbf{v}, \tag{5.48}$$

where $B(\alpha)$ is according to the definition of consistently ordered (2.17).
Thus $\mathbf{v}$ is an eigenvector of $B(\sqrt{\mu})$ with eigenvalue

$$\lambda = \frac{\mu+\omega-1}{\sqrt{\mu}\omega}. \tag{5.49}$$

Since A is consistently ordered, it follows that λ is also an eigenvalue of B_{Jac}.

A similar argument can be made in the reverse direction, showing that each eigenvalue λ of B_{Jac} corresponds to an eigenvalue μ of B_{SOR}. □

Relation (5.45) allows to connect the spectral radii of the iteration matrices. In particular, setting $\omega = 1$ (corresponding to the Gauss-Seidel method), we obtain

$$\lambda = \frac{\mu}{\sqrt{\mu}} \quad \Longrightarrow \quad \rho(B_{\text{GS}}) = \rho(B_{\text{Jac}})^2, \tag{5.50}$$

which implies that the Gauss-Seidel method converges approximately twice as fast as the Jacobi method in terms of the spectral radius [20, sec.5.5].

Note that, by Theorem 2.6, all matrices derived in Chapter 3 are consistently ordered. Therefore, relation (5.45) holds for these matrices.

5.3. Comparison Iterative Methods

In this section, we compare the results of the three iterative methods: Jacobi, Gauss-Seidel (GS), and Successive Over-Relaxation (SOR).

We aim to approximate the function

$$f(x) = \sin(\pi x), \tag{5.51}$$

on an interval $[a, b]$, which we determine depending on the boundary condition whose second derivative is

$$f''(x) = \pi^2 \sin(\pi x). \tag{5.52}$$

This function satisfies Dirichlet boundary conditions at points $x = m$, where $m \in \mathbb{Z}$, In this setup, we use $a = 0$, $b = 1$ for the Dirichlet case.

For Neumann boundary conditions, which occur at $x = 0.5 + m$, we choose $a = 0.5$, and $b = 1.5$. For mixed boundary conditions, we use $a = 0$ and $b = 0.5$.

For all iterative methods we run until stopping criteria from Section 5.1 are met with tolerance of 10^{-6}, or when the maximum number of iterations has been reached. We chose this to be $500 \cdot n$ iterations, where n is the dimension of the matrix.

Table 5.1: Number of iterations required using relative difference and relative residual as stopping criteria for various boundary conditions and solution methods. If the relative difference or residual was not small enough to meet the stopping criterion, the final value is shown in parentheses. Text colours indicate RMSE (see Section 2.5) quality: green = low (good), orange = moderate, red = high (poor).

		$n = 100$		$n = 500$		$n = 1000$	
Boundary Condition	Method	Relative difference	Relative residual	Relative difference	Relative residual	Relative difference	Relative residual
Dirichlet boundary conditions, Interval: $[0, 1]$							
	Jacobi	12782	28555	154026	$(7.34 \cdot 10^{-3})$	361258	$(8.52 \cdot 10^{-2})$
	GS (L)	7107	14279	94019	$(5.38 \cdot 10^{-5})$	242050	$(7.26 \cdot 10^{-3})$
	GS (U)	7107	14279	94019	$(5.38 \cdot 10^{-5})$	242050	$(7.26 \cdot 10^{-3})$
	SOR	1062	1515	966	1643	3584	6498
Neumann boundary conditions, Interval: $[0.5, 1.5]$							
	Jacobi	12782	28555	154026	$(7.34 \cdot 10^{-3})$	361258	$(8.52 \cdot 10^{-2})$
	GS (L)	7107	14300	94019	$(5.40 \cdot 10^{-5})$	242050	$(7.28 \cdot 10^{-3})$
	GS (U)	7107	14300	94019	$(5.40 \cdot 10^{-5})$	242050	$(7.28 \cdot 10^{-3})$
	SOR	890	1461	953	1795	3644	7694
Neumann boundary conditions (symmetrized), Interval: $[0.5, 1.5]$							
	Jacobi	$(7.08 \cdot 10^{-6})$	$(1.43 \cdot 10^{-2})$	153258	$(7.60 \cdot 10^{-3})$	360492	$(8.44 \cdot 10^{-2})$
	GS (L)	6868	13729	94367	$(4.98 \cdot 10^{-5})$	241451	$(7.13 \cdot 10^{-3})$
	GS (U)	6868	13729	94367	$(4.98 \cdot 10^{-5})$	241451	$(7.13 \cdot 10^{-3})$
	SOR	876	1435	948	1762	3631	7350
Mixed boundary conditions, Interval: $[0, 0.5]$							
	Jacobi	39718	$(2.36 \cdot 10^{-3})$	$(2.03 \cdot 10^{-6})$	$(2.93 \cdot 10^{-1})$	$(1.44 \cdot 10^{-6})$	$(5.40 \cdot 10^{-1})$
	GS (L)	22708	$(5.59 \cdot 10^{-6})$	242349	$(8.56 \cdot 10^{-2})$	$(1.02 \cdot 10^{-6})$	$(2.92 \cdot 10^{-1})$
	GS (U)	22709	$(5.70 \cdot 10^{-6})$	242349	$(8.60 \cdot 10^{-2})$	$(1.02 \cdot 10^{-6})$	$(2.93 \cdot 10^{-1})$
	SOR	958	1657	3573	7486	12440	31803
Mixed boundary conditions (symmetrized), Interval: $[0, 0.5]$							
	Jacobi	39096	$(2.01 \cdot 10^{-3})$	$(2.03 \cdot 10^{-6})$	$(2.91 \cdot 10^{-1})$	$(1.45 \cdot 10^{-6})$	$(5.40 \cdot 10^{-1})$
	GS (L)	22341	$(4.43 \cdot 10^{-6})$	241750	$(8.50 \cdot 10^{-2})$	$(1.01 \cdot 10^{-6})$	$(2.92 \cdot 10^{-1})$
	GS (U)	22341	$(4.39 \cdot 10^{-6})$	241750	$(8.48 \cdot 10^{-2})$	$(1.01 \cdot 10^{-6})$	$(2.91 \cdot 10^{-1})$
	SOR	951	1601	3560	7152	12419	30426

Note: In the case of a singular matrix (e.g., when two Neumann boundary conditions are applied), the linear system $A\mathbf{w} = \mathbf{f}$ may still be solvable. However, the resulting solution is not necessarily the true solution of the original differential problem. It is one of potentially infinitely many solutions to the consistent system.

For $n = 100$, all iterative solvers converge successfully across all boundary conditions and associated matrix structures. The relative residuals and relative differences achieved are consistently small, indicating reliable approximations of the solution. Notably, the SOR method converges with significantly fewer iterations than both the Jacobi and Gauss-Seidel (GS) methods, demonstrating superior efficiency even for small system sizes. However, for the symmetrized versions of the Neumann and mixed boundary condition matrices, the performance of the iterative solvers is somewhat diminished, requiring more iterations to reach acceptable error thresholds. An example of the final result for Dirichlet boundary conditions for $n = 100$ is illustrated in Figure 5.2.

As the problem size increases to $n = 500$, notable differences in the performance of the methods emerge. The Jacobi method fails to meet the prescribed stopping criterion across all boundary condition types, with relative residuals stagnating around 10^{-2}, which exceeds typical tolerances for numerical accuracy. In contrast, the GS method continues to converge for Dirichlet and Neumann boundary conditions; however, it fails for cases involving mixed boundary conditions, where the relative residuals plateau between 0.08 and 0.09. Even in successful cases, the GS method produces residuals close to the tolerance threshold of 10^{-5}. Notably, the SOR method maintains consistently strong performance at $n = 500$, converging in all tested scenarios with significantly fewer iterations. The visualisation of the final result of the behaviour for mixed boundary conditions is shown in Figure 5.3.

For $n = 1000$, the differences in method performance become even more pronounced. Both the Jacobi and GS methods fail to meet the stopping criterion for all boundary conditions. In these cases, the relative residuals increase substantially, reaching values as high as 0.54. The GS method, in particular, stagnates at residuals in the range of 0.07 to 0.09, indicating an inability to reduce the error through iterative refinement within the maximum number of iterations. In contrast, the SOR method continues to converge reliably in all problem instances. While the number of iterations naturally increases with n, SOR consistently shows efficient convergence behaviour.

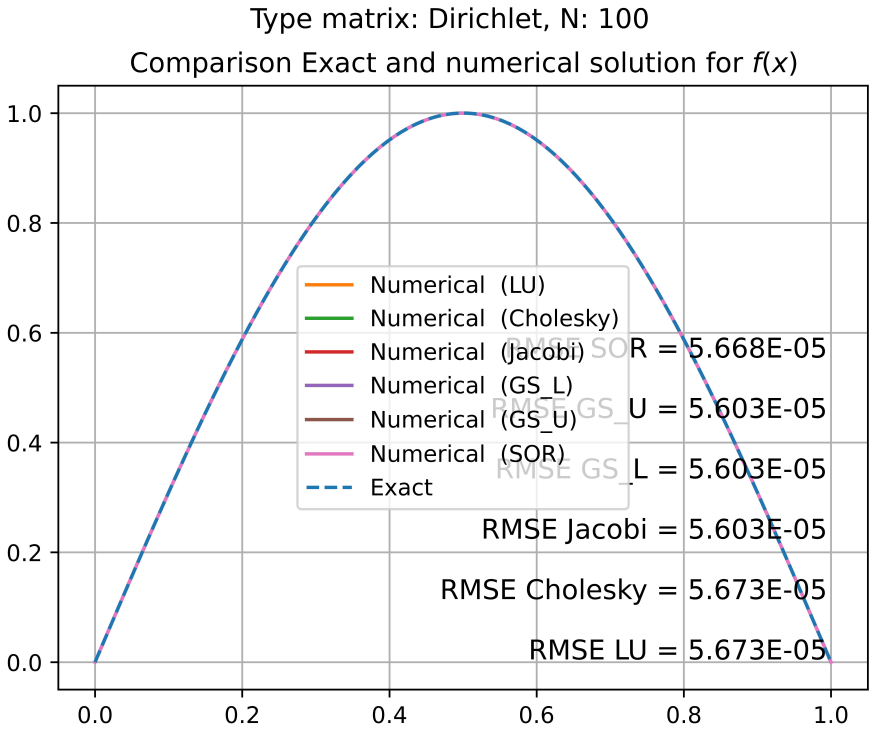


Figure 5.2: Convergence for Dirichlet boundary condition with $n = 100$. All iterative methods converge to satisfy the stopping criteria.

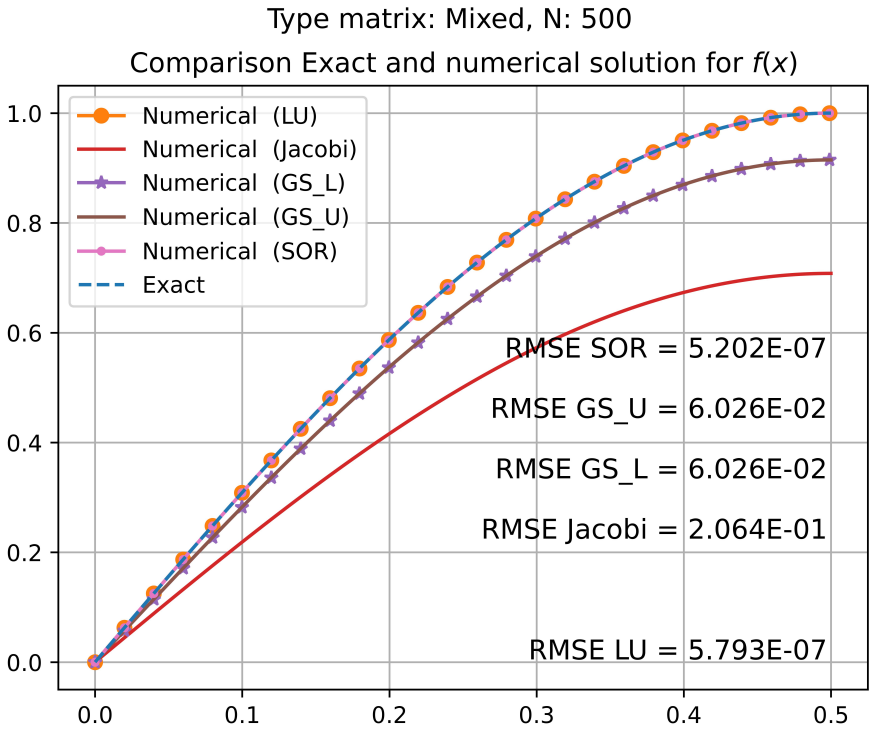


Figure 5.3: Convergence for mixed boundary conditions with $n = 500$. Jacobi and GS methods stagnate before meeting the stopping criterion; final residuals remain above tolerance.

6

Convergence Properties

6.1. Order of Convergence

We evaluate the accuracy of the obtained approximated solution and measure the order of convergence of the different discretization matrices and numerical methods. For this, we approximate the function $\sin(\pi x)$, and solve the related systems as defined in Chapter 5.

We perform simulations using grids with $n = 20, 40, 80, 160$ and 320 interior points. On each grid, we solve the discretized problem and compare the computed solution u_h to the exact solution $u(x) = \sin(\pi x)$. A quantitive number for the correctness of the solution is given using the root mean square error (see Section 2.5). This time again, the maximum number of iterations is $500 \cdot n$.

To determine the empirical order of convergence p, we consider two consecutive grids with n_1 and n_2 interior points. Let dx_1 and dx_2 be their corresponding grid sizes, i.e.

$$dx_i = \frac{b-a}{n+1}, \quad i = 1, 2. \tag{6.1}$$

Then we compute, using Richardson extrapolation [23, sec. 1.3],

$$p \approx \frac{\log(\text{RMSE}(dx_1)/\text{RMSE}(dx_2))}{\log(dx_1/dx_2)}. \tag{6.2}$$

Table 6.1 summarizes the observed orders of convergence ρ for various discretization methods under different boundary conditions.

As shown in Table 6.1, most methods exhibit a consistent second-order convergence rate, particularly for Dirichlet boundary conditions. In contrast, for some combinations of methods and boundary conditions, most notably the Jacobi method under Neumann, mixed, and symmetrized mixed boundary conditions, the order becomes negative. These negative values are because the iterative solver did not converge within the maximum allowed number of iterations, set at $500 \cdot n$. Therefore, the stopping criteria did not guarantee a reasonable good approximation.

LU and Cholesky methods (where applicable) have reliable and consistent convergence rates, while SOR and Gauss-Seidel (both lower and upper variants) generally perform well. The Jacobi method performs well for small problem sizes.

Table 6.1: Empirical order of convergence for various boundary conditions and solution methods. Each column labeled $a \to b$ shows the observed order of convergence when refining the mesh from a to b internal grid points. Missing values correspond to methods that are not applicable for the specified boundary condition.

Boundary condition	**Method**	**20 → 40**	**40 → 80**	**80 → 160**	**160 → 320**	**Expected**
Dirichlet	LU	2.0	2.0	2.0	2.0	2.0
	Cholesky	2.0	2.0	2.0	2.0	
	Jacobi	2.0	2.0	2.0	-3.9	
	GS_L	2.0	2.0	2.0	2.1	
	GS_U	2.0	2.0	2.0	2.1	
	SOR	2.0	2.0	2.0	2.0	
Neumann	LU	-	-	-	-	2.0
	Cholesky	-	-	-	-	
	Jacobi	2.0	2.0	2.0	-3.9	
	GS_L	2.0	2.0	2.0	2.1	
	GS_U	2.0	2.0	2.0	2.1	
	SOR	2.0	2.0	2.0	2.0	
Neumann symmetric	LU	-	-	-	-	1.0
	Cholesky	-	-	-	-	
	Jacobi	1.0	1.0	1.0	0.9	
	GS_L	1.0	1.0	1.0	1.0	
	GS_U	1.0	1.0	1.0	1.0	
	SOR	1.0	1.0	1.0	1.0	
Mixed	LU	2.0	2.0	2.0	2.0	2.0
	Cholesky	-	-	-	-	
	Jacobi	2.0	-2.1	-5.5	-2.7	
	GS_L	2.0	2.0	-4.0	-5.5	
	GS_U	2.0	2.0	-4.0	-5.5	
	SOR	1.9	2.0	2.0	2.0	
Mixed symmetric	LU	1.0	1.0	1.0	1.0	1.0
	Cholesky	1.0	1.0	1.0	1.0	
	Jacobi	1.0	0.9	-1.1	-2.5	
	GS_L	1.0	1.0	0.9	-1.9	
	GS_U	1.0	1.0	0.9	-1.9	
	SOR	1.0	1.0	1.0	1.0	

6.2. Spectral Radius

As established in Equation (5.14), an iterative method converges for all initial guesses and right-hand sides if and only if the spectral radius $\rho(B) < 1$, where B is the iteration matrix. To assess convergence behaviour for different boundary condition types and iteration methods, we numerically compute $\rho(B)$ as the maximum absolute value of the eigenvalues of B.

Figures 6.1a to 6.3b show the value of $1 - \rho(B)$ for systems with Dirichlet, Neumann, symmetrized Neumann, mixed, and symmetrized mixed boundary conditions, respectively. This quantity emphasizes how close the spectral radius is to the critical value of 1.

Convergent Cases ($\rho(B) < 1$): For the Dirichlet and both mixed boundary condition cases (standard and symmetrized), $1 - \rho(B)$ is consistently positive. At high resolutions (e.g., $n = 1000$), this difference lies between 10^{-5} and 10^{-6}, indicating convergence at a moderate rate.

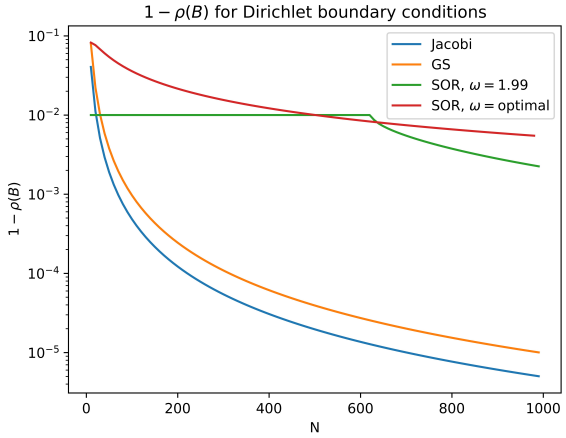
Among all methods:

- The Jacobi method consistently exhibits the largest spectral radius (i.e., slowest convergence).
- Gauss-Seidel (GS) improves upon Jacobi, as expected from the inequality in Equation (5.50).
- Successive Over-Relaxation (SOR) with optimal ω achieves the smallest spectral radius (i.e., fastest convergence). However, for small systems, SOR with fixed (suboptimal) $\omega = 1.99$ may perform slightly worse than GS. This is expected since the optimal ω depends on the problem size and matrix characteristics.
- A clear intersection between the curves for $\omega = 1.99$ and optimal ω is visible, indicating that $\omega_{\text{opt}} = 1.99$ for that specific problem size.

Non-Convergent Cases ($\rho(B) \approx 1$): For both Neumann cases (standard and symmetrized), the computed spectral radius differs from 1 only by 10^{-14} to 10^{-16}, which lies within floating-point precision. This suggests $\rho(B) = 1$, and thus, no convergence is expected according to Equation (5.14).

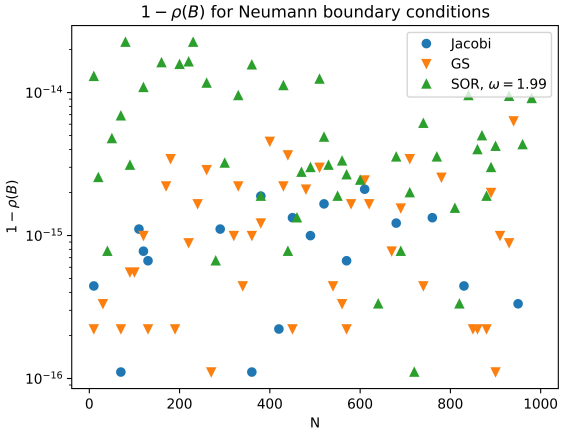
In these cases, $\rho_{\text{Jac}} \approx 1$, making the standard formula for computing the optimal relaxation parameter ω in SOR (Equation (5.40)) inapplicable, as it leads to a division by zero.

Despite this, the convergence behaviour for the Neumann cases appears similar to the Dirichlet case. This can be explained by the eigenvalue spectrum: only two eigenvalues are very close to zero, while the remainder closely resembles the Dirichlet spectrum. Figure 6.4 illustrates this by comparing the full spectrum and its deviation from 1 for $n = 200$.

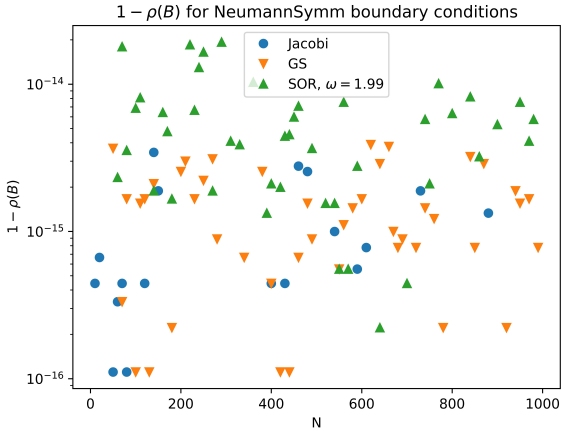


(a) $1-\rho(B)$ for Dirichlet boundary conditions.

Figure 6.1: Spectral radius of iteration matrix for system with Dirichlet boundary conditions.

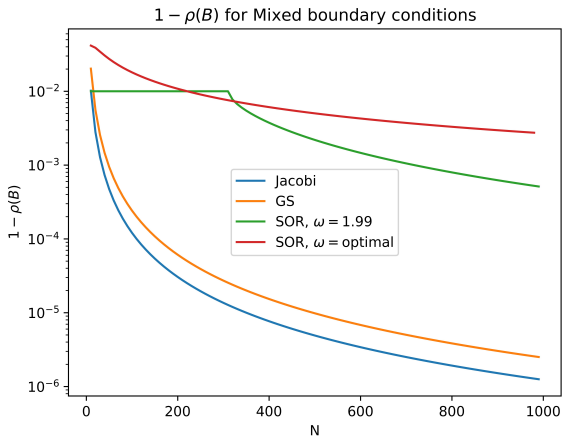


(a) $1-\rho(B)$ for Neumann boundary conditions.

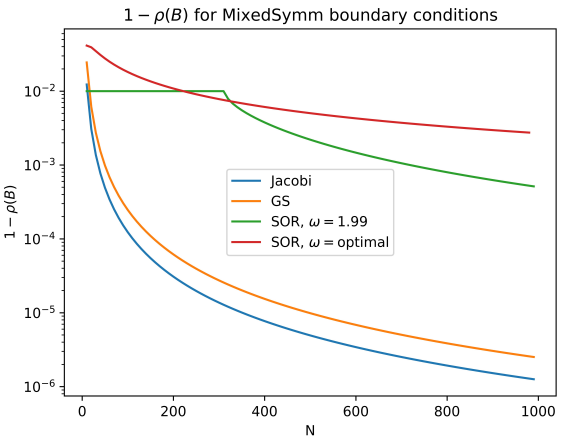


(b) $1-\rho(B)$ for symmetrized Neumann boundary conditions.

Figure 6.2: Spectral radius of iteration matrix for systems with Neumann boundary conditions.

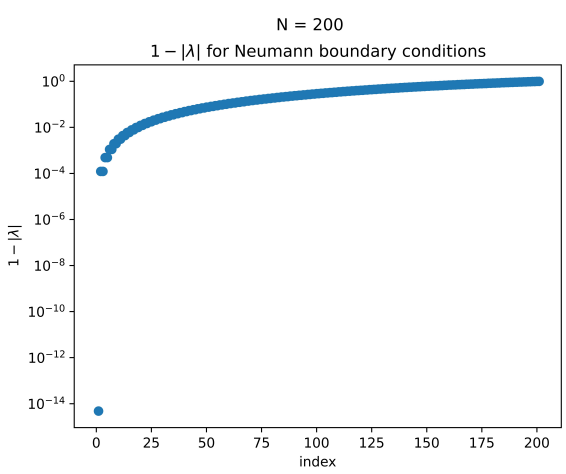


(a) $1-\rho(B)$ for mixed boundary conditions.

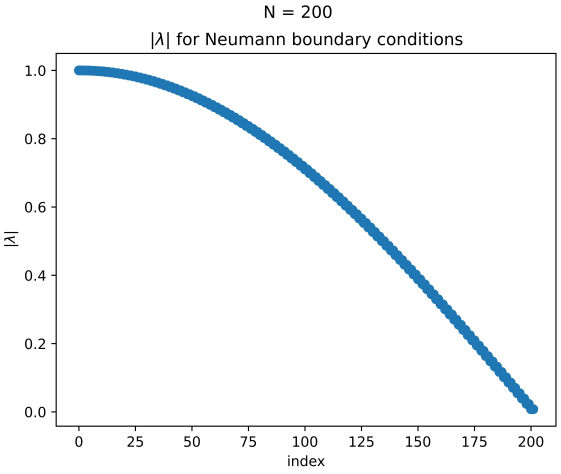


(b) $1-\rho(B)$ for symmetrized mixed boundary conditions.

Figure 6.3: Spectral radius of iteration matrix for systems with mixed boundary conditions.



(a) $1-|\lambda|$ of spectrum for Neumann case, $n = 200$.



(b) $|\lambda|$ of spectrum for Neumann case, $n = 200$.

Figure 6.4: Spectral distribution of the iteration matrix for Neumann boundary conditions. Only a few eigenvalues deviate significantly.

7

Conclusion

The central aim of this research was to investigate: *How does the structure of system matrices affect the behaviour of direct and iterative solvers in terms of speed and accuracy?*

Throughout this study, we examined several boundary condition scenarios, namely Dirichlet, Neumann, and mixed, in both their standard and symmetrized forms, and analysed the properties of the resulting system matrices. These structural properties directly influenced the performance and applicability of both direct and iterative solution methods.

From a theoretical perspective, we found that:

- Matrices resulting from Dirichlet and Mixed boundary conditions are non-singular.
- Matrices from the Dirichlet and Mixed (symmetrized) cases are positive definite.
- The symmetrized Neumann matrix is positive semi-definite, and the standard Neumann case results in a singular matrix.

These properties determine the feasibility of using specific solvers.

For direct solvers, LU decomposition consistently succeeded on all non-singular matrices. Cholesky decomposition, however, was only applicable when the matrix was symmetric and positive definite. This confirms theoretical expectations and highlights that Cholesky, while more efficient when applicable, has stricter requirements than LU. Both methods delivered highly accurate solutions.

For iterative solvers, convergence was largely determined by the spectral radius $\rho(B)$ of the iteration matrix. As predicted by theory, convergence is guaranteed when $\rho(B) < 1$. For systems with Dirichlet and mixed (including the symmetrized versions), this condition was met, and convergence was observed, especially for Gauss-Seidel and SOR methods.

Although Neumann systems resulted in matrices with spectral radius very close to 1, suggesting theoretical non-convergence, iterative solvers still showed convergence in practice. This apparent contradiction can be explained by the eigenvalue spectrum: only a small number of eigenvalues were near zero, while the rest closely resembled the spectra of well-conditioned systems. These eigenvalues controlled the convergence behaviour, enabling the solver to converge effectively despite theoretical limitations.

Furthermore, the spectral radius analysis confirmed several theoretical relationships:

- Gauss-Seidel consistently outperformed Jacobi in terms of convergence speed. In the case of tridiagonal systems, the relation $\rho(B_{\text{GS}}) = \rho(B_{\text{Jac}})^2$ was clearly observed.
- Among the tested iterative methods, SOR yielded the fastest convergence when the optimal relaxation parameter ω_{opt} was used. In practical applications, however, this optimal value is generally unknown,

as its computation requires prior knowledge of the spectral radius of the Jacobi iteration matrix, precisely the quantity one aims to avoid computing.

- For large systems, the Jacobi method, and eventually also Gauss-Seidel, exhibited noticeably slow convergence. Their performance diminished significantly compared to SOR, particularly when a high level of accuracy was desired within a limited number of iterations.

In summary, the structure of the system matrix, particularly its singularity, symmetry, and definiteness, plays a crucial role in determining the viability and efficiency of both direct and iterative solvers. While direct methods offer robustness and accuracy, iterative methods provide speed and scalability, provided that spectral properties are favourable.

8

Discussion

This thesis has investigated the impact of matrix structure, induced by different boundary conditions, on the performance of both direct and iterative solvers for linear systems. The results confirmed theoretical expectations regarding convergence criteria, spectral radius behavior, and method applicability. Nevertheless, several limitations and open questions remain.

Limitations and Scope

The analysis was restricted to one-dimensional Laplacian problems with specific boundary conditions. While this setting is sufficient for illustrating key concepts and validating theoretical predictions, it does not fully capture the complexity of real-world systems, especially in higher dimensions or in the presence of more complex physics.

Furthermore, the study focused on classical solvers such as LU decomposition, Cholesky factorization, Jacobi, Gauss-Seidel, and SOR. While these are foundational, more advanced methods (e.g., multigrid, preconditioned Krylov methods) were outside the scope of this work.

Additionally, although the spectral radius served as a useful convergence indicator in most cases, it proved insufficient for explaining convergence behavior in singular systems, such as those with Neumann boundary conditions. There, the spectral radius was effectively equal to 1, yet the iterative methods still exhibited convergence due to the structure of the remaining spectrum. This calls for a more refined theoretical treatment of such cases.

More Singular Systems

An important observation is that even small perturbations toward singularity, such as a nearly-zero diagonal element caused by discretization or modeling assumptions, can severely degrade the performance of both direct and iterative methods. In such cases:

- Direct solvers may become numerically unstable, especially LU decomposition without pivoting, as small pivots lead to large rounding errors.
- Iterative solvers become unreliable since the near-zero diagonal values impair diagonal dominance, a key condition for convergence of Jacobi and Gauss-Seidel methods.
- SOR methods with fixed relaxation parameters may completely fail or even diverge, as optimal ω becomes highly sensitive or undefined.

This sensitivity highlights the importance of matrix conditioning. While theoretical convergence criteria often assume ideal properties (e.g., strict diagonal dominance or positive definiteness), practical discretizations may violate these assumptions, resulting in degraded solver behaviour or even complete failure.

Suggestions for Future Work

To build on this study, several research directions are proposed:

- Higher-dimensional problems: Extending the analysis to two- or three-dimensional Laplacians would reveal how structural complexity and sparsity patterns affect solver performance at scale.
- General PDEs: Analyzing more diverse equations, such as time-dependent, nonlinear, or convection-diffusion problems, would improve understanding of solver behavior in broader applications.
- Sharper theoretical estimates: Numerical results suggest that existing bounds (e.g., for spectral radii) may be overly conservative. Tighter estimates could enhance predictive power and guide solver selection.
- Convergence in singular systems: As shown, $\rho(B) < 1$ does not always capture convergence for singular systems. Alternative metrics may yield more comprehensive convergence criteria.
- Robustness to near-singularity: Further study of how discretization artifacts affect system conditioning could inform regularization techniques or adaptive solver strategies to mitigate instability.

These avenues offer opportunities to strengthen both the theoretical foundations and practical reliability of numerical solvers in increasingly realistic settings.

Bibliography

[1] Parsiad Azimzadeh. A fast and stable test to check if a weakly diagonally dominant matrix is a nonsingular M-matrix. *Mathematics of Computation*, 88(316):783–800, 2018. ISSN 1088-6842. doi: https://doi.org/10.1090/mcom/3347.

[2] Anthony G. Barnston. Correspondence among the Correlation, RMSE, and Heidke Forecast Verification Measures; Refinement of the Heidke Score. *Weather and Forecasting*, 7(4):699–709, 1992. ISSN 0882-8156. URL https://doi.org/10.1175/1520-0434(1992)007<0699:CATCRA>2.0.CO;2.

[3] Michele Benzi. Preconditioning Techniques for Large Linear Systems: A Survey. *Journal of Computational Physics*, 182(2):418–477, 2002. ISSN 0021-9991. URL https://doi.org/10.1006/jcph.2002.7176.

[4] Neil V. Budko. Lecture 3. Finite Difference Method in 1D, order of approximation, properties of 1D FD Laplacian, 2024.

[5] James W. Demmel. *Applied Numerical Linear Algebra*. Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics, 1997. ISBN 978-0-89871-389-3. URL https://doi.org/10.1137/1.9781611971446.

[6] John B. Fraleigh and Raymond A. Beauregard. *Linear Algebra*. Pearson Education Limited, 3 edition, 2014. ISBN 978-1-292-04272-5.

[7] Gene H. Golub and Charles F. Van Loan. *Matrix Computations - 4th Edition*. Johns Hopkins University Press, 2013. ISBN 978-1-4214-0794-4. URL https://doi.org/10.1137/1.9781421407944.

[8] Wolfgang Hackbusch. *Iterative Solution of Large Sparse Systems of Equations*. Applied Mathematical Sciences. Springer Cham, 2 edition, 2016. ISBN 978-3-319-28483-5. URL https://doi.org/10.1007/978-3-319-28483-5.

[9] Nicholas J. Higham. *Analysis of the Cholesky decomposition of a semi-definite matrix*, pages 161–186. Oxford University PressOxford, 1990. URL https://doi.org/10.1093/oso/9780198535645.003.0010.

[10] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, 2 edition, 2012. ISBN 9780521839402. URL https://doi.org/10.1017/CBO9781139020411.

[11] J. van Kan, A. Segal, K. Vermolen, and H. Kraaijevanger. *Classical Numerical Methods in Scientific Computing*. Delft Academic Press, 2023. ISBN 978-94-6366-731-9. URL https://doi.org/10.59490/t.2023.007.

[12] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. Society for Industrial and Applied Mathematics, 1995. ISBN 978-0898713527. URL https://doi.org/10.1137/1.9781611970944.

[13] Randall J. LeVeque. *Finite Difference Methods for Ordinary and Partial Differential Equations*. Society for Industrial and Applied Mathematics, 2007. ISBN 978-0-89871-629-0. URL https://doi.org/10.1137/1.9780898717839.

[14] Carl D. Meyer. *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics, 2000. ISBN 9780898714548.

[15] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. *Numerical Mathematics*. Texts in Applied Mathematics. Springer Berlin, Heidelberg, 2 edition, 2010. ISBN 978-3-540-49809-4. doi: https://doi.org/10.1007/b98885.

[16] Rolf Rannacher. *Numerical Linear Algebra*. Lecture Notes. Heidelberg University Publishing, 2018. ISBN 978-3-946054-99-3. URL https://doi.org/10.17885/heiup.407.

[17] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, 2003. ISBN 978-0-89871-534-7. URL https://doi.org/10.1137/1.9780898718003.

[18] Richard S. Varga. *Successive Overrelaxation Iterative Methods*, pages 97–131. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-05156-2. URL https://doi.org/10.1007/978-3-642-05156-2_4.

[19] C. Vuik. *Iterative solution methods*. Delft Institute of Applied Mathematics, 2025.

[20] C. Vuik and D.J.P. Lahaye. *Scientific Computing (wi4201)*. Delft Institute of Applied Mathematics, 2019.

[21] C. Vuik, F.J. Vermolen, M.B. van Gijzen, and M.J. Vuik. *Numerical Methods for Ordinary Differential Equations*. Delft Academic Press, 2nd edition, 2016. ISBN 97890-6562-3737. URL https://doi.org/10.5074/t.2023.001.

[22] David Young. Iterative Methods for Solving Partial Difference Equations of Elliptic Type. *Transactions of the American Mathematical Society*, 76(1):92–111, 1954. ISSN 00029947, 10886850. doi: 10.2307/1990745. URL https://doi.org/10.1090/S0002-9947-1954-0059635-7.

[23] Zahari Zlatev, Ivan Dimov, István Faragó, and Ágnes Havasi. *Richardson Extrapolation*, volume 2. De Gruyter, 2018. ISBN 978-3-11-053300-2.

A

LU Decomposition with Pivoting

In practice, straightforward LU decomposition as described in Section 4.2.1 may fail or become numerically unstable when the matrix A has small or zero pivot elements. To address this, pivoting strategies are employed to improve stability and guarantee the existence of the factorization for any nonsingular matrix.

Pivoting involves rearranging the rows of A via a permutation matrix P, so that the factorization takes the form

$$PA = LU, \tag{A.1}$$

where L is lower triangular with unit diagonal entries, U is upper triangular, and P encodes the row swaps.

The most common approach is partial pivoting, which selects the largest pivot element by absolute value in each column to minimize rounding errors during elimination.

Algorithm 8 details the PLU decomposition with partial pivoting. At each step i, the algorithm searches for the pivot row $p \geq i$ with the largest absolute value in column i, then swaps rows i and p in U and the permutation matrix P. If any elimination has been done in previous columns, the corresponding rows of L are also swapped to maintain consistency.
After pivoting, Gaussian elimination proceeds to zero out entries below the pivot, storing multipliers in L.

Once the factorization $PA = LU$ is obtained, the linear system

$$A\mathbf{w} = \mathbf{f} \tag{A.2}$$

can be solved by first applying the permutation,

$$P\mathbf{f} = \tilde{\mathbf{f}}, \tag{A.3}$$

then performing forward substitution to solve

$$L\mathbf{y} = \tilde{\mathbf{f}}, \tag{A.4}$$

and finally backward substitution to solve

$$U\mathbf{w} = \mathbf{y}. \tag{A.5}$$

This procedure is detailed in Algorithm 9 [6, sec. 10.3].

Algorithm 8 PLU Decomposition (with Partial Pivoting)

Initialize P as identity matrix of size $n \times n$
Initialize L as identity matrix of size $n \times n$
Initialize U as the same matrix as A (of size $n \times n$)
for $i = 1$ to n **do**
    Find index $p \geq i$ such that $\left|U[p][i]\right|$ is maximal
    if $p \neq i$ **then**
        Swap rows i and p in U
        Swap rows i and p in P
        if $i > 1$ **then**
            Swap rows i and p in the first $i-1$ columns of L
        end if
    end if
    pivot $\leftarrow U[i][i]$
    for $j = i+1$ to n **do**
        $c \leftarrow \dfrac{U[j][i]}{\text{pivot}}$
        $L[j][i] \leftarrow c$
        for $k = 1$ to n **do**
            $U[j][k] \leftarrow U[j][k] - c \cdot U[i][k]$
        end for
    end for
    $L[i][i] \leftarrow 1$
end for
return L, P, U

Algorithm 9 Solve $A\mathbf{w} = \mathbf{f}$ using PLU decomposition

Require: Lower triangular matrix L, upper triangular matrix U,
Permutation matrix P, right-hand side vector $\mathbf{b}$

Apply permutation: $\tilde{\mathbf{f}} \leftarrow P\mathbf{f}$
Forward substitution: solve $L\mathbf{y} = \tilde{\mathbf{f}}$
for $i = 1$ to n **do**
    $y[i] \leftarrow \tilde{f}[i] - \sum_{j=1}^{i-1} L[i][j] \cdot y[j]$
end for
Back substitution: solve $U\mathbf{w} = \mathbf{y}$
for $i = n$ down to 1 **do**
    $w[i] \leftarrow \dfrac{1}{U[i][i]} \left(y[i] - \sum_{j=i+1}^{n} U[i][j] \cdot w[j] \right)$
end for
return x

B

Source Code

The Python source code for matrix construction, RMSE computation, and both direct and iterative solvers is available on GitHub:
https://github.com/jtdenhertog/Solution-Methods-in-Numerical-Linear-Algebra.