Solving the steady-state heat equation using overdetermined non-overlapping domain decomposition methods

Filip Thor

Supervisors

Claus Führer
Philipp Birken

Examiner

Eskil Hansen

Abstract

Domain decomposition methods can be used to numerically solve partial differential equations for certain problems, for example in cases where the domain has an irregular shape, or if there are differences in material constants. By splitting the domain into subdomains, these problems can be solved using domain decomposition methods. In this thesis, the topic is solving the steady-state heat equation using more than one boundary condition for each subdomain, causing the domain decomposition method to be overdetermined. The least squares method is used to handle this, and so it is explored if, by modifying the method to use parts of the mathematical formulation as constraints, the method will find an adequate approximation to the steady-state heat equation. It was found that overdetermined domain decomposition methods can indeed find a good approximation of the temperature distribution, and that using a constrained least squares method with different types of relaxation, can decrease the number of iterations to reach termination. This paves way for more work in relation to the use of overdetermined domain decomposition methods.

Keywords: Overdetermined non-overlapping domain decomposition methods, constrained least squares methods, steady-state heat equation

Acknowledgements

I would like to begin by thanking my supervisors, Claus Führer and Philipp Birken, for their exceptional assistance and guidance throughout this effort. A special thanks to Claus who has taken several evenings out of his private time to assist in the finalisation of the thesis. I would also like to thank Gabrielle Flood, Azahar Monge Sánchez, Ivo Dravins, and Lea Versbach for their assistance in answering questions and listening to the issues that arose during the work effort. Additionally, I want to thank Line Elmshøj for the help of reading through the thesis, and to Niklas Hansson, Emma Sjöborg and Johannes Olsson who has offered vast amounts of support. Finally, I would like to thank my family for the their curiosity, interest, help, and love. I am very grateful.

Thank you.

Contents

Chapter 1

Introduction

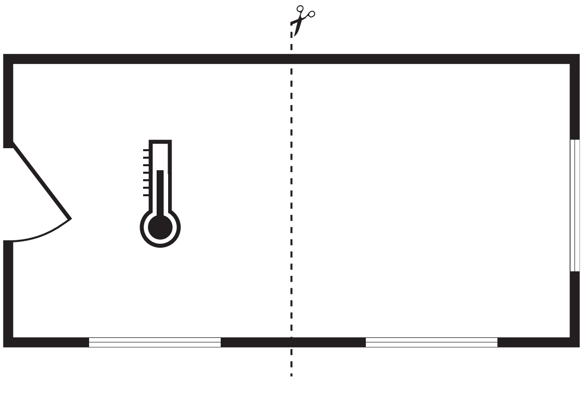
1.1 Background

Imagine that you are building a one-room house, and you have decided that you want to fit in a radiator to ensure that the room is comfortably warm. There is a large window, and a small air vent, letting air seep in. How large does the radiator have to be? What capacity does it have to have? You could purchase one based on a guess, but you run the risk of buying something that does not satisfy your wish for a pleasantly warm room. Thankfully, it is possible to calculate the heat distribution in the room using mathematics.

The room has four walls, a window, an air vent and your radiator. There are different ways of expressing these things mathematically. The walls, the radiator, and the windows can be viewed as having a constant temperature. The walls and windows enclose the room, and are what is referred to as a *boundary* of the room. The room itself is called a *domain*. When a boundary has a fixed value, in our case a fixed temperature, this is called a *Dirichlet* boundary condition. The air vent letting in a constant stream of cold air is distinctly different than the walls or window. The flow of cold air is referred to as the flux, and for the boundary where the air vent is located, there is what is referred to as a *Neumann* boundary condition.

Throughout this thesis, the example room used is a simplified. The room has two sides which act like radiators, and two sides which act like widows.

Finding the temperature in a room such as the case above is, given enough simplification, can be done by paper and pen, finding a continuous temperature distribution. However, in most cases it becomes almost impossible, and for those cases, an approximation of the temperature in the room might be enough. This is generally done by finding the temperature in specific points in the room, but it comes at a cost. The more points you are looking for, the longer it takes to compute. In some cases it might be enough to get a general idea of what the temperature looks like, whereas just a few points are necessary. In other cases, where you might have many small sources of heat, using few points might result in your model not picking up all the sources, and you end up with a result that does not adequately represent the reality.



Instead of looking at finding the temperature in the entire room at once, it is possible to split the room into two parts, and finding the temperature for each part separately. With our example case, it would be perfectly fine to keep it as one domain and solve it as such. In more complex problems however, it might be very difficult, or require a lot of simplification to solve as one domain. It is in these cases that splitting the domain into smaller pieces might help in finding a good solution.

Finding the temperature in each part requires that the boundary between the parts, called an *interface*, to be known. This is not the case, so a guess has to be made. With this guess, a solution can be produced for each part. From these solutions, a better estimation of the interface is produced. By repeatedly solving parts of the room, the estimation for the interface gets better and better. This iterative approach eventually sees a decline in improvement of the estimation. When the change of the interface is very small between iterations, the solution produced by the two parts of the room is a good approximation of the temperature distribution in the entire room. Methods such as these are called *domain decomposition methods.*

In splitting the room into two parts, it is necessary to mathematically describe the new boundary along the interface for each part. The most common method for this is stipulating that for one part, the boundary has a fixed temperature — a Dirichlet condition, whilst the other has a fixed inflow, or flux — a Neumann condition. This method is known as the Dirichlet-Neumann method.

Domain decomposition methods have proven to be very advantageous in areas such as continuum mechanics [4], fluid dynamics [3], and more specifically, heating and cooling in different materials [2].

1.2 Aims

I will focus on the temperature distribution over a room as a case study, by looking at a different way of formulating the boundary between the two parts, or *subdomains.* The idea is to formulate two boundary conditions for the interface for each subdomain — both a Dirichlet- and a Neumann condition.

When finding an approximate solution for the entire room as the one described above, it is a matter of solving a system of linear equations. In splitting the room in two parts, and adding a boundary condition along the interface, the system of linear equations is solveable, since there are the same number of equations as there are unknowns.

However, in adding more than one condition along the interface, the system becomes overdetermined. One method to deal with this is to use the least squares method. This method distributes the deviation, or *residual* for each unknown evenly over all unknowns. It is possible to set constraints as to how the deviation is distributed, using a so called constrained least squares method. I will investigate if the least squares method using boundary conditions as constraints improves upon the method.

To summarise, the aims of this thesis is to investigate —

- whether a domain decomposition method with more than one boundary condition per interface can find a good approximation to the temperature distribution,
- how to handle the overdetermined nature of a discrete domain decomposition method with more than one boundary condition,
- whether modifying the least squares method for solving overdetermined discrete problems work worse, as good as, or better than currently existing domain decomposition methods,
- how these modifications affect the methods.

1.3 Delimitations

Since the spectrum of problems and methods one can divulge in is large, a number of choices have been made in order to keep the scope narrow — many of the options mentioned here are further discussed in

Chapter **7. Future Work**.

There are many different ways to decide how the points for which the temperature is found. In this thesis, I have limited myself to use a regular, equidistant grid of points. The discretisation used is the second order central difference scheme.
One type of problem has been solved, finding the temperature distribution in a rectangular room without windows, where two walls are warm and two are cold. The temperature distribution when the temperature has settled to equilibrium, referred to as the *steady-state* temperature distribution, is sought.

In short,

— the effect of different grid sizes has not been evaluated,

— one type of problem has been evaluated, the steady-state heat equation over a rectangle room,

— two subdomains have been formulated, which are side by side,

— one type of discretisation scheme has been used, the second-order finite differences discretisation scheme,

— the problem is formulated as a multi-domain formulation, and not on Schur complement form [3] (p. 52).

1.4 Methodology

I will start by formulating the discretised Laplace equation over the entire room, as a reference when validating the overdetermined methods.

By dividing the room into subdomains, I will formulate the mathematical tools needed to then set up the problem to be solved with the Dirichlet-Neumann method, a commonly used domain decomposition method, to use as a comparison when evaluating how well my own methods work.

Next follows a part where I construct my own methods, where I introduce both a Dirichlet-, and a Neumann condition on both subdomains along the interface, and propose different approaches on how to deal with the overdetermined nature of the systems of linear equations that arise.

The findings are presented at the end of the thesis, along with a discussion and ideas fo future work.

Chapter 2

The Global Problem Formulation

2.1 Problem formulation

The problem on which the methods will be used upon remains the same throughout the thesis — finding the steady-state temperature distribution over a room. The room is rectangular, with the long sides twice as long as the short sides. There are no windows, two of the walls are warm, and two are cold. The north and west walls (boundaries ∂_{N} and ∂_{W}) have temperature 30°C, and the south and east walls (boundaries ∂_{S} and ∂_{W}) have temperature 15°C, as illustrated in the figure below.

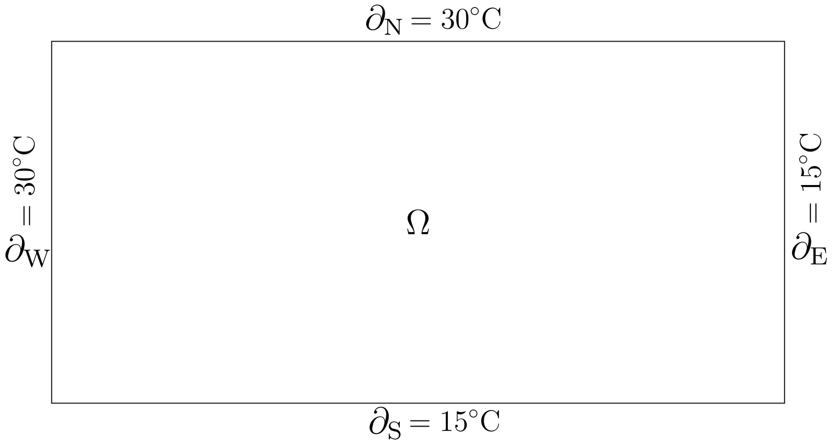

$\partial_{\mathrm{N}} = 30°\mathrm{C}$
$\partial_{\mathrm{W}} = 30°\mathrm{C}$
Ω
$\partial_{\mathrm{E}} = 15°\mathrm{C}$
$\partial_{\mathrm{S}} = 15°\mathrm{C}$

Figure 2.1: The domain and its boundary conditions.

The solution produced by the discretised Laplace equation is used as a reference solution $\mathbf{u^r}$. The grid remains the same as to be able to calculate the aberration between the reference solution and the solutions produced from my own methods. This reference solution can be seen in Figure 2.2.

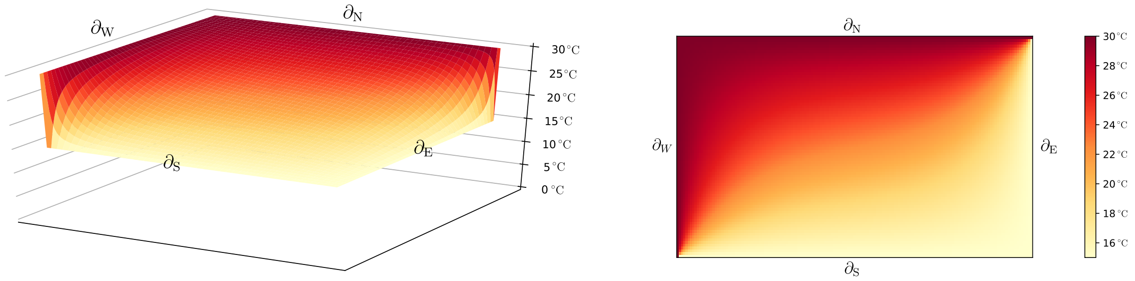
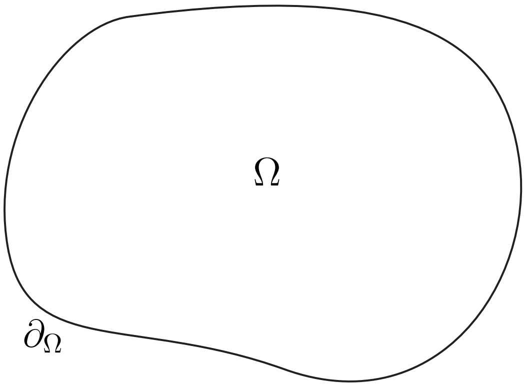


Figure 2.2: The reference temperature distribution over the room.

2.2 Finding the Approximate Temperature Distribution



Finding the steady-state temperature distribution of an area can be done by solving the Laplace equation over the domain Ω. $u(x, y)$ is the function describing the temperature distribution within the domain. Since there is no source of heat inside the domain, only the surroundings affect the temperature distribution. The surrounding area is expressed as a boundary around the domain, ∂_Ω which encloses the domain. The conditions over the boundary are expressed by the function g. For our example, the walls surrounding the room are expressed as a Dirichlet condition with a given temperature.

$$\begin{aligned} \Delta u(x, y) &= 0 \quad \text{in } \Omega \in \mathrm{R}^2 \\ u &= g \quad \text{on } \partial_\Omega. \end{aligned} \tag{2.1}$$

Solving this equation yields the steady-state temperature distribution over the domain Ω.

In order to find an approximate solution $u(x_i, y_j)$, we discretise the equation and solve it for the discrete points (x_i, y_j) on a grid over the room.

The grid

The room is twice as long as it is wide. Along the width of the room are placed n discrete grid points. For simplicity later on in the thesis, the parts of the room consists of two squares size $n \times n$, with the interface between. This means that the room consists of $n \times (2n + 1)$ internal points.

These internal points $u(x_i, y_j) = u_{i,j}$ are gathered in a column vector $\mathbf{u}$, starting from the left top corner as the first element $(1, 1)$, going along the first row, then the second row, and so on;

$$\mathbf{u} = \begin{pmatrix} u_{1,1} \\ u_{1,2} \\ \vdots \\ \hline u_{2,1} \\ \vdots \\ \hline \vdots \\ \hline \vdots \\ u_{n,2n+1} \end{pmatrix} \in \mathbb{R}^{2n^2+n} \tag{2.2}$$

where the last element is the bottom right element. The walls are also gathered as column vectors $\mathbf{b}_\mathrm{N}$, $\mathbf{b}_\mathrm{W}$, $\mathbf{b}_\mathrm{S}$, and $\mathbf{b}_\mathrm{E}$, the long walls having $2n+1$ elements, and the short walls n elements, as seen in Figure 2.3.

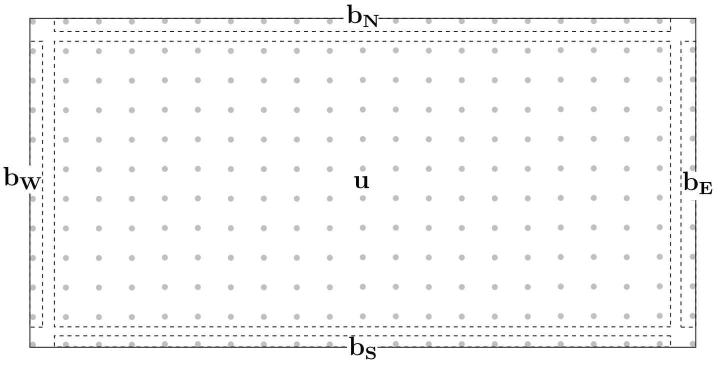


Figure 2.3: An illustration of the domain and the grid, with the different vectors indicated.

The discretisation

Next, we discretise the Laplace equation using a second order central difference scheme,

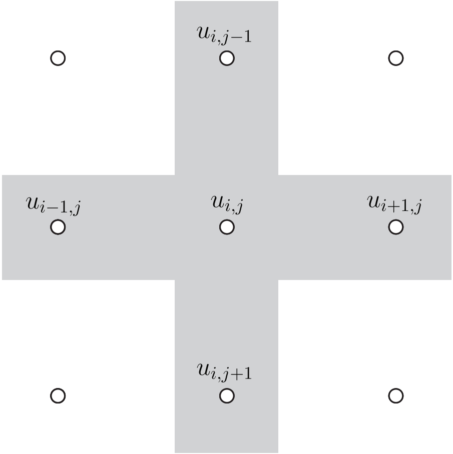
$$\Delta u(x_i, y_j) \approx \frac{u_{i+1,j} + u_{i-1,j} - 4u_{i,j} + u_{i,j+1} + u_{i,j-1}}{\Delta x^2} \tag{2.3}$$

where Δx is the distance between each grid point, given by

$$\frac{L}{n} = \Delta x,$$

where L is the length of the short wall of the room. This thesis deals with the Laplace equation, $\Delta u(x_i, y_j) = 0$ which means that in the equations that arise from the discretisation, Δx can be disregarded.

The second order central difference discretisation can be visualised as a dot stencil, as seen in the illustration below,



which illustrates the manner in which the temperature in a grid point is influenced by the temperature in grid points in its vicinity. For internal points the unknowns follow the form given in Equation (2.3);

$$u_{i-1,j} + u_{i+1,j} - 4u_{i,j} + u_{i,j-1} + u_{i,j+1} = 0.$$

For grid points right next to a wall, the known temperatures in those points enter as viewed in Figure 2.4.

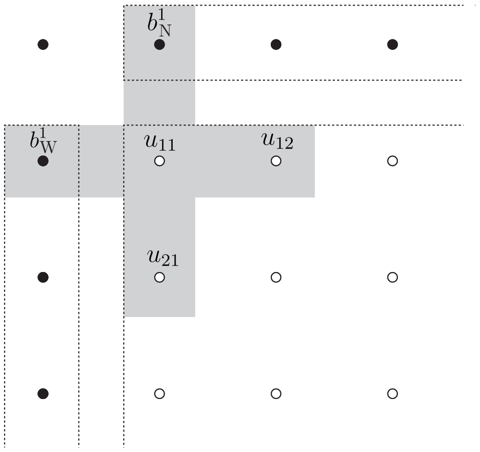


Figure 2.4: The dot stencil for unknowns adjacent to the boundary.

Starting from the first element, $(1,1)$ in Equation (2.3), one equation is obtained,

$$b_{\mathrm{W}}^{1} - 4u_{11} + u_{12} + b_{\mathrm{N}}^{1} + u_{21} = 0.$$

Doing the same for each point in that row, continuing with all rows, gives a system of linear equations with one equation for each unknown;

$$\begin{aligned} b_{\mathrm{W}}^{1} - 4u_{11} + u_{12} + b_{\mathrm{N}}^{1} + u_{21} &= 0 \\ u_{11} - 4u_{12} + u_{13} + b_{\mathrm{N}}^{2} + u_{22} &= 0 \\ u_{12} - 4u_{13} + u_{14} + b_{\mathrm{N}}^{3} + u_{23} &= 0 \\ &\vdots \end{aligned}$$

The discrete values along the boundaries enter as indicated above. These values are known, and are therefore moved to the right-hand side of the equality sign.

$$\begin{aligned} -4u_{11} + u_{12} + u_{21} &= -b_{\mathrm{N}}^{1} - b_{\mathrm{W}}^{1} \\ u_{11} - 4u_{12} + u_{13} + u_{22} &= -b_{\mathrm{N}}^{2} \\ u_{12} - 4u_{13} + u_{14} + u_{23} &= -b_{\mathrm{N}}^{3} \\ &\vdots \end{aligned}$$

This system of linear equations can be expressed by matrices. The unknowns $u_{i,j}$ are arranged as a vector $\mathbf{u}$, introduced in Equation (2.2). Forming a block matrix $\mathbf{A} \in \mathbb{R}^{2n^2+n \times 2n^2+n}$, and having all knowns, (the values that enter from the boundary) in a column vector $\mathbf{b}$, the system of linear equations is now expressed by matrices as

$$\mathbf{Au} = \mathbf{b}.$$

The characteristics of $\mathbf{b}$ is quite straightforward. The characteristics of $\mathbf{A}$, is a bit more intricate.

A consists of n^2 blocks of size $2n + 1 \times 2n + 1$ because there are $n \times (2n + 1)$ internal points in the grid. This is due to the structure of **u** according to Equation (2.2).

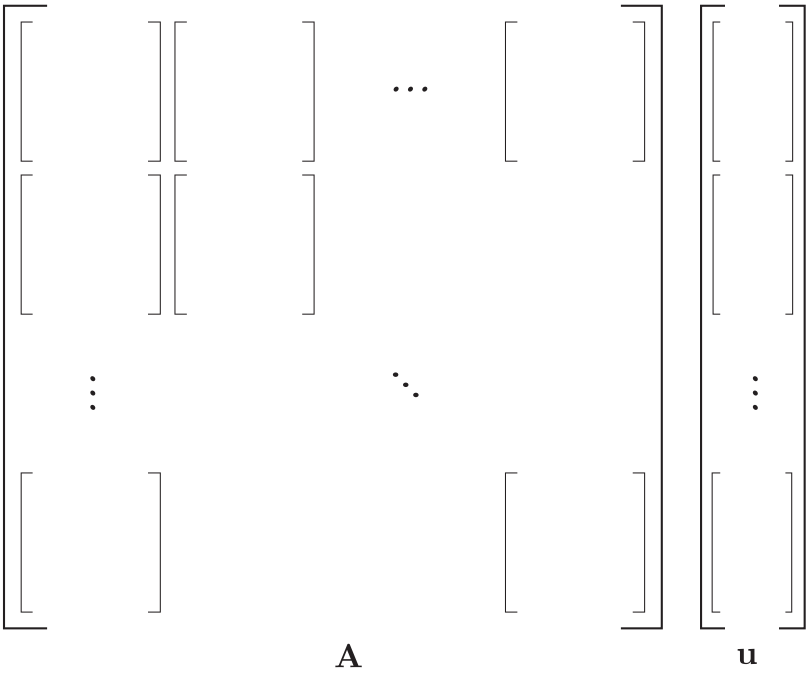


Figure 2.5: Graphical illustration of **A** and **u**. **A** consists of n^2 blocks, where each block has $(2n + 1) \times (2n + 1)$ elements.

Starting from the first row of block matrices in **A**, the first block details the discretisation of the Laplace equation over the grid points on the same row in the discretisation. The second block in the same row details the influence from the second row of grid points. There is no influence from the third row on the first row of grid points according to the discrete Laplace equation, so that block will be zeroes only, and the same follows for all blocks that row. For the first row of grid points, the sets of linear equations can be grouped as follows;

$$
\begin{array}{cccc}
\begin{array}{r} \text{element (1,1):} \\ \text{element (1,2):} \\ \text{element (1,3):} \\ \vdots \end{array}
&
\underbrace{\begin{array}{cccc} -4u_{11} & +u_{12} & & \\ u_{11} & -4u_{12} & +u_{13} & \\ & u_{12} & -4u_{13} & +u_{14} \\ & \vdots & & \end{array}}_{\text{One } 2n+1 \times 2n+1 \text{ block, accessing elements from the same row}}
&
\underbrace{\begin{array}{ccc} +u_{21} & & \\ & +u_{22} & \\ & & +u_{23} \\ & & \vdots \end{array}}_{\text{Accessing elements from the row immediately below}}
&
\begin{array}{cl} = & -b^1_{\mathrm{N}} - b^1_{\mathrm{W}} \\ = & -b^2_{\mathrm{N}} \\ = & -b^3_{\mathrm{N}} \\ \vdots & \end{array} .
\end{array}
$$

Followng the same structure, each row of the block matrix has a similar structure. From Row 2 the four point stencil will access both grid points from the row below and above the current grid point:

$$
\begin{array}{cccc}
\begin{array}{r} \text{element (2,1):} \\ \text{element (2,2):} \\ \text{element (2,3):} \\ \vdots \end{array}
&
\underbrace{\begin{array}{ccc} u_{11} & & \\ & u_{12} & \\ & & u_{13} \\ & \vdots & \end{array}}_{\text{Accessing elements from the row immediately above}}
&
\underbrace{\begin{array}{cccc} -4u_{21} & +u_{22} & & \\ +u_{21} & -4u_{22} & +u_{23} & \\ & +u_{22} & -4u_{23} & +u_{24} \\ & \vdots & & \end{array}}_{\text{Accessing elements from the same row}}
&
\underbrace{\begin{array}{ccc} +u_{31} & & \\ & +u_{32} & \\ & & +u_{33} \\ & \vdots & \end{array}}_{\text{Accessing elements from the row immediately below}}
\end{array}
$$

After covering all rows of grid points in the discretisation, the assembled block matrix **A** gets the structure:

$$\mathbf{A} = \left(\begin{array}{cccc|cccc|c}
-4 & 1 & & 0 & 1 & & & & \\
1 & -4 & \ddots & & & 1 & & & 0 \\
 & \ddots & \ddots & 1 & & & \ddots & & \\
0 & & 1 & -4 & 0 & & & 1 & \\
\hline
1 & & & 0 & -4 & 1 & & 0 & \ddots \\
 & 1 & & & 1 & -4 & \ddots & & \\
 & & \ddots & & & \ddots & \ddots & 1 & \\
 & & & 1 & 0 & & 1 & -4 & \\
\hline
 & 0 & & & \ddots & & & & \ddots
\end{array}\right) \tag{2.4}$$

The solution from solving this system of linear equation is used as a reference solution, $\mathbf{u}^r$.

2.3 The Global case, formulated as two subdomains

2.3.1 The general case

So far the temperature distribution is found for the domain as a whole. Now, the domain is split into subdomains Ω_1 and Ω_2 with the interface Γ inbetween. Essentially, this is simply a reshuffling in which order the equations are in. By doing so, it is possible to express the necessary matrices to be used later on.

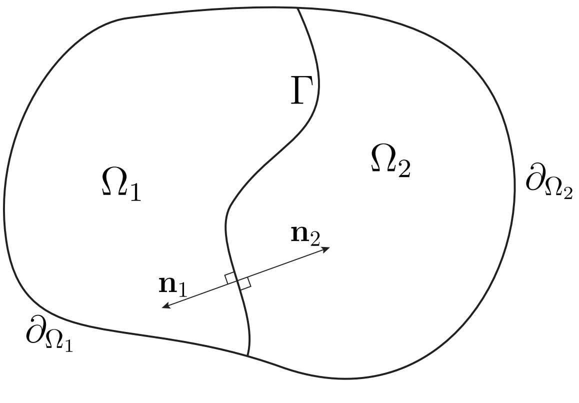


Figure 2.6: Splitting a domain into two nonoverlapping subdomains

By splitting up the initial domain into subdomains, special conditions have to be formulated to guarantee continuity over the interface boundary Γ [4]. The conditions are the following:

$$\Omega_1 : \begin{cases} \Delta u_1 = 0 \quad \text{in } \Omega_1 & i) \text{ The Laplace equation is applied to the internal points in } \Omega_1. \\ u_1 = g \quad \text{on } \partial_{\Omega_1} & ii) \text{ The values along the boundary } \partial_{\Omega_1} \text{ are given by } g. \end{cases} \tag{2.5}$$

$$\Gamma : \begin{cases} u_1 = u_2 \quad \text{on } \Gamma & i) \text{ The values along the interface } \Gamma \text{ have to be the same.} \\ \frac{\partial u_1}{\partial \mathbf{n}_1} = -\frac{\partial u_2}{\partial \mathbf{n}_2} \quad \text{on } \Gamma & ii) \text{ The values of the derivatives over the interface } \Gamma \text{ have to be the same.} \end{cases} \tag{2.6}$$

$$\Omega_2 : \begin{cases} \Delta u_2 = 0 \quad \text{in } \Omega_2 & i) \text{ The Laplace equation is applied to the internal points in } \Omega_2. \\ u_2 = g \quad \text{on } \partial_{\Omega_2} & ii) \text{ The values along the boundary } \partial_{\Omega_2} \text{ are given by } g. \end{cases} \tag{2.7}$$

This is now done on the example problem of this thesis:

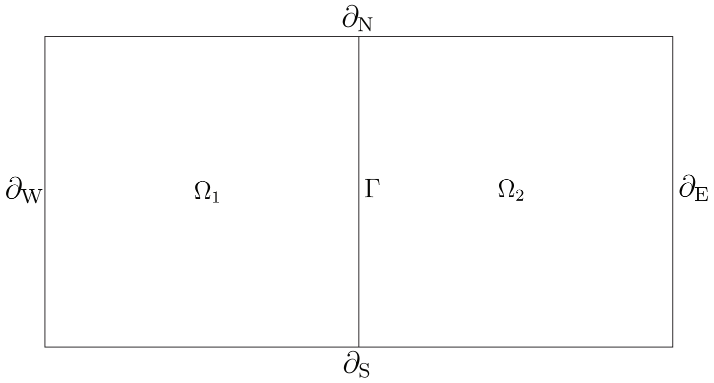


Figure 2.7: A schematic overview of the domain and the subdomains Ω_1 and Ω_2, with the interface Γ inbetween.

2.3.2 Discretising the global case as two subdomains

Due to the shape of the room, it is easily split into two equal sized parts, where subdomain Ω_1, Ω_2 consists of $n \times n$ internal points and interface Γ having n points:

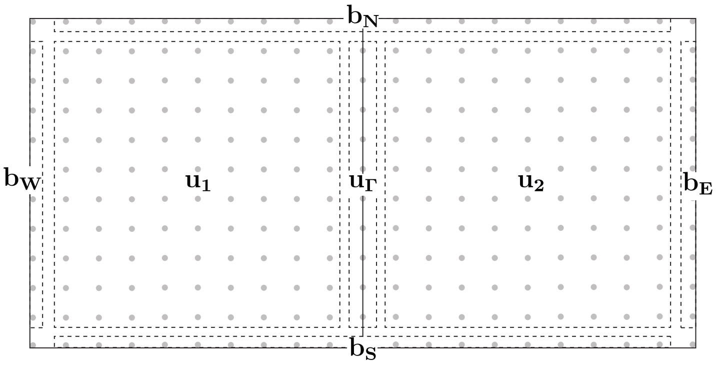


Figure 2.8: A schematic overview of the discretised subdomains.

The internal points are now arranged separately from each other as three vectors. Two vectors $\mathbf{u_1}$ and

$\mathbf{u_1}$ for the subdomains and $\mathbf{u_\Gamma}$ for the common interface,

$$\mathbf{u_1} = \left(\begin{array}{c} u_1^{11} \\ u_1^{12} \\ u_1^{13} \\ \vdots \\ \hline u_1^{21} \\ \vdots \\ \hline \vdots \\ \hline u_1^{n1} \\ \vdots \\ u_1^{nn} \end{array}\right) \in \mathbb{R}^{n^2}, \quad \mathbf{u_2} = \left(\begin{array}{c} u_2^{11} \\ u_2^{12} \\ u_2^{13} \\ \vdots \\ \hline u_2^{21} \\ \vdots \\ \hline \vdots \\ \hline u_2^{n1} \\ \vdots \\ u_2^{nn} \end{array}\right) \in \mathbb{R}^{n^2}, \quad \mathbf{u_\Gamma} = \left(\begin{array}{c} u_\Gamma^{1} \\ u_\Gamma^{2} \\ u_\Gamma^{3} \\ \vdots \\ \\ u_\Gamma^{n} \end{array}\right) \in \mathbb{R}^{n}.$$

Partitioning the grid points for each subdomain and its interface, each set of grid points has its own set of conditions which are handled separately from each other. These conditions are the ones found in Equations (2.5), (2.6), and (2.7).

Conditions over Ω_1

$$\Omega_1 : \begin{cases} \Delta u_1 = 0 \quad \text{in } \Omega_1 & i) \quad \text{The Laplace equation is applied to the internal points in } \Omega_1. \\ u_1 = g \quad \text{on } \partial_{\Omega_1} & ii) \quad \text{The values along the boundary } \partial_{\Omega_1} \text{ are given by } g. \end{cases}$$

When employing each grid point as described in Equation (2.3), the process is similar to that explained in detail earlier, in **2.2 Finding the Approximate Temperature Distribution**, except for the last point on each row, where a grid point from the interface enters:

$$b_w^1 - 4u_1^{1,n} + \boxed{u_\Gamma^1} + b_\mathrm{N}^n + u_1^{2,n} = 0$$

The same is true for all grid points adjacent to the interface. By creating another block matrix $\mathbf{A_{1\Gamma}}$ with n blocks (one block for each row) of size $n \times n$. Each block of $\mathbf{A_{1\Gamma}}$ affects each row of grid points, each row within the blocks affects each grid point within that row. The grid points at the end of each row are adjacent to the interface, so in $\mathbf{A_{1\Gamma}}$ all elements are zero except for the bottom row of each block, where the coefficient is 1 for the grid point adjacent to the row in subdomain Ω_1.

For all other grid points in each row of the subdomain, the equations remain the same as previously mentioned, although the set of grid points in $\mathbf{u_1}$ is fewer, since only the grid points in the first half of the room is expressed. The block matrix containing the coefficients for $\mathbf{u_1}$ — called $\mathbf{A_{11}}$ — then has to be smaller. $\mathbf{A_{11}}$ is similar to $\mathbf{A}$, also a block matrix, but with n^2 blocks of size $n \times n$. The values entering from the boundary in a similar fashion (as they are known) moved to the right-hand side of the equality sign, which gives the discretisation for Ω_1 the following system of linear equations:

$$
\underbrace{\left(\begin{array}{cccc|cccc|c}
-4 & 1 & & 0 & 1 & & & & \\
1 & -4 & \ddots & & & 1 & & & \\
 & \ddots & \ddots & 1 & & & \ddots & & \\
0 & & 1 & -4 & 0 & & & 1 & \\
\hline
1 & & & 0 & -4 & 1 & & & \ddots \\
 & 1 & & & 1 & -4 & \ddots & & \\
 & & \ddots & & & \ddots & \ddots & 1 & \\
 & & & 1 & & & 1 & -4 & \\
\hline
 & & & & \ddots & & & & \ddots
\end{array}\right)}_{\mathbf{A_{11}}}
\underbrace{\left(\begin{array}{c}
u_1^{11} \\ u_1^{12} \\ u_1^{13} \\ \vdots \\ \hline u_1^{21} \\ \vdots \\ \\ \hline \vdots \\ \hline u_1^{n1} \\ \vdots \\ u_1^{nn}
\end{array}\right)}_{\mathbf{u_1}}
+
\underbrace{\left(\begin{array}{cccc}
\vdots & & & \\ 0 & & & \\ 1 & 0 & \dots & \\ \hline \vdots & & & \\ 0 & 1 & 0 & \dots \\ \hline & \ddots & & \\ \hline & & & \\ & & & \vdots \\ & \dots & 0 & 1
\end{array}\right)}_{\mathbf{A_{1\Gamma}}}
\underbrace{\left(\begin{array}{c}
u_\Gamma^1 \\ u_\Gamma^2 \\ u_\Gamma^3 \\ \vdots \\ \\ u_\Gamma^n
\end{array}\right)}_{\mathbf{u_\Gamma}}
=
\underbrace{\left(\begin{array}{c}
-b_{\mathrm{N}}^1 - b_{\mathrm{W}}^1 \\ -b_{\mathrm{N}}^2 \\ -b_{\mathrm{N}}^3 \\ \vdots \\ \hline -b_{\mathrm{W}}^2 \\ 0 \\ \vdots \\ \hline -b_{\mathrm{W}}^3 \\ 0 \\ \vdots \\ \hline \vdots \\ \hline -b_{\mathrm{W}}^n - b_{\mathrm{S}}^1 \\ -b_{\mathrm{S}}^2 \\ \vdots
\end{array}\right)}_{\mathbf{b_1}}
$$

which can neatly be summarised as

$$\mathbf{A_{11}u_1} + \mathbf{A_{1\Gamma}u_\Gamma} = \mathbf{b_1}. \tag{2.8}$$

Conditions over Ω_2

$$
\Omega_2 : \begin{cases}
\Delta u_2 = 0 \quad \text{in } \Omega_2 & \textit{i)} \quad \text{The Laplace equation is applied to the internal points in } \Omega_2. \\
u_2 = g \quad \text{on } \partial_{\Omega_2} & \textit{ii)} \quad \text{The values along the boundary } \partial_{\Omega_2} \text{ are given by } g.
\end{cases}
$$

The same approach is applied to the second subdomain. However, for this case, the interface is on the left-hand side of the subdomain. For this, a matrix is needed to access the adjacent grid point in the interface only for the points at the start of each row — $\mathbf{A_{2\Gamma}}$. It has the same size as $\mathbf{A_{1\Gamma}}$. Since the first grid point in each row is adjacent to the interface, only these equations are influenced by the interface. Therefore, in each block of $\mathbf{A_{2\Gamma}}$, all elements are zero except for the top row, where each coefficient is 1 for the respective interface grid point.

$$
\underbrace{\left(\begin{array}{ccc}
1 & 0 & \dots \\ 0 & & \\ \vdots & & \\ \hline 0 & 1 & 0\dots \\ \vdots & & \\ & & \\ \hline & \ddots & \\ \hline \dots & 0 & 1 \\ & & 0 \\ & & \vdots
\end{array}\right)}_{\mathbf{A_{2\Gamma}}}
\mathbf{u_\Gamma} +
\underbrace{\left(\begin{array}{cccc|cccc|c}
-4 & 1 & & 0 & 1 & & & & \\
1 & -4 & \ddots & & & 1 & & & \\
 & \ddots & \ddots & 1 & & & \ddots & & \\
0 & & 1 & -4 & 0 & & & 1 & \\
\hline
1 & & & 0 & -4 & 1 & & & \ddots \\
 & 1 & & & 1 & -4 & \ddots & & \\
 & & \ddots & & & \ddots & \ddots & 1 & \\
 & & & 1 & & & 1 & -4 & \\
\hline
 & & & & \ddots & & & & \ddots
\end{array}\right)}_{\mathbf{A_{22}}}
\underbrace{\left(\begin{array}{c}
u_2^{11} \\ u_2^{12} \\ u_2^{13} \\ \vdots \\ \hline u_2^{21} \\ \vdots \\ \hline \vdots \\ \hline u_2^{n1} \\ \vdots \\ u_2^{nn}
\end{array}\right)}_{\mathbf{u_2}}
=
\underbrace{\left(\begin{array}{c}
\vdots \\ -b_{\mathrm{N}}^{2n-1} \\ -b_{\mathrm{N}}^{2n} \\ -b_{\mathrm{N}}^{2n+1} - b_{\mathrm{E}}^1 \\ \hline \vdots \\ 0 \\ -b_{\mathrm{E}}^2 \\ \hline \vdots \\ 0 \\ -b_{\mathrm{E}}^2 \\ \hline \vdots \\ \hline \vdots \\ -b_{\mathrm{E}}^{2n} \\ -b_{\mathrm{E}}^{2n+1} - b_{\mathrm{E}}^n
\end{array}\right)}_{\mathbf{b_2}}
$$

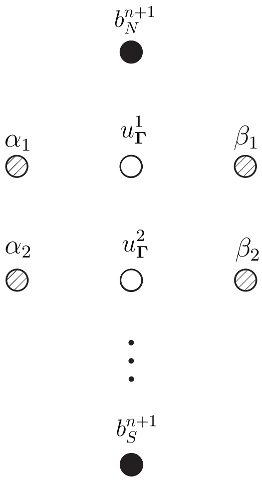
which is summarised as

$$\mathbf{A_{2\Gamma}u_\Gamma} + \mathbf{A_{22}u_2} = \mathbf{b_2}. \tag{2.9}$$

Conditions over Γ

$$\Gamma : \begin{cases} u_1 = u_2 & \text{on } \Gamma \quad i) \quad \text{The values along the interface } \Gamma \text{ have to be the same.} \\ \frac{\partial u_1}{\partial \mathbf{n}_1} = -\frac{\partial u_2}{\partial \mathbf{n}_2} & \text{on } \Gamma \quad ii) \quad \text{The derivative over the boundary } \Gamma \text{ have to be the same incoming as outgoing.} \end{cases} \tag{2.10}$$

The second condition says that the inflow and outflow over the interface have to be equal. This is done by imposing a Neumann condition along Γ. This is done by adding a set of unknowns, α_i, and β_i. Applying the second order central difference discretisation for each grid point gives us a set of equations. In this case there are two points from the walls that enter; b_{N}^{n+1} and b_{S}^{n+1}.



$$\begin{aligned} \alpha_1 - 4u_{\mathbf{\Gamma}}^1 + \beta_1 + b_{\text{N}}^{n+1} + u_{\mathbf{\Gamma}}^2 &= 0 \\ \alpha_2 - 4u_{\mathbf{\Gamma}}^2 + \beta_2 + u_{\mathbf{\Gamma}}^1 + u_{\mathbf{\Gamma}}^3 &= 0 \\ \alpha_3 - 4u_{\mathbf{\Gamma}}^3 + \beta_3 + u_{\mathbf{\Gamma}}^2 + u_{\mathbf{\Gamma}}^4 &= 0 \\ &\vdots \\ \alpha_n - 4u_{\mathbf{\Gamma}}^n + \beta_n + u_{\mathbf{\Gamma}}^{n-1} + b_{\text{S}}^{n+1} &= 0 \end{aligned} \tag{2.11}$$

The second condition in Equation (2.10) is implemented using a first-order forward difference scheme [3] (p.4), the unknowns α_i and β_i can be expressed using the grid points from each subdomain.

The inflow to Ω_1 is

$$\frac{\partial u_1}{\partial \mathbf{n}_1},$$

the inflow to Ω_2 is

$$\frac{\partial u_2}{\partial \mathbf{n}_2}.$$

Expressing these using the first order forward difference scheme, we have that the derivatives are

$$\frac{\partial u_1}{\partial \mathbf{n}_1} = \frac{u_{\mathbf{\Gamma}}^i - \alpha_i}{\Delta x}$$

and

$$\frac{\partial u_2}{\partial \mathbf{n}_2} = \frac{\beta_i - u_{\mathbf{\Gamma}}^i}{\Delta x}.$$

In order to highlight which values are where, some more detailed notation needs to be introduced. $u_{\mathbf{1}}^{i,n}$ denotes all values from $\mathbf{u_1}$ with indices {(1,n), (2,n), (3,n), ..., (n,n) }, *i.e* the grid points in $\mathbf{u_1}$ adjacent to the interface. Similarly for $\mathbf{u_2}$, the grid point adjacent to the interface have the indices {(1,1), (2,1), (3,1), ..., (n,*1*) }

The inflow should be equal to the outflow, which can also be expressed using the first order first difference scheme over the subdomains. The outflow from Ω_2 is

$$-\frac{\partial u_2}{\partial \mathbf{n}_2} = -\frac{u_{\mathbf{2}}^{i,1} - u_{\mathbf{\Gamma}}^i}{\Delta x}$$

and the outflow from Ω_1 is

$$-\frac{\partial u_1}{\partial \mathbf{n}_1} = -\frac{u_{\mathbf{\Gamma}}^i - u_{\mathbf{1}}^{i,n}}{\Delta x}.$$

As the condition states, the inflow and the outflow should be equal. So, we find that the inflow to Ω_1 from Ω_2 is

$$\frac{\partial u_1}{\partial \mathbf{n}_1} = -\frac{\partial u_2}{\partial \mathbf{n}_2}$$
$$\Downarrow$$
$$\frac{u_{\mathbf{\Gamma}}^{i} - \alpha_i}{\Delta x} = -\frac{u_{\mathbf{2}}^{i,1} - u_{\mathbf{\Gamma}}^{i}}{\Delta x} \Rightarrow \alpha_i = u_{\mathbf{2}}^{i,1}$$

and the inflow to Ω_2 from Ω_2 is

$$\frac{\partial u_2}{\partial \mathbf{n}_2} = -\frac{\partial u_1}{\partial \mathbf{n}_1}$$
$$\Downarrow$$
$$\frac{\beta_i - u_{\mathbf{\Gamma}}^{i}}{\Delta x} = -\frac{u_{\mathbf{\Gamma}}^{i} - u_{\mathbf{1}}^{i,n}}{\Delta x} \Rightarrow \beta_i = u_{\mathbf{1}}^{i,n}.$$

In a similar way as $\mathbf{A_{1\Gamma}}$ accesses the interface grid points next to the subdomains for their respective equations, transposing the matrix does the opposite, instead accessing the grid poins from $\mathbf{u_1}$ which are adjacent to the interface grid points. This matrix is called $\mathbf{A_{\Gamma 1}}$.

In a similar fashion, for the grid points in $\mathbf{u_2}$ adjacent to the interface are the grid points with indices {(1,1), (2,1), (3,1), ..., (n,1)}, expressed as $u_{\mathbf{2}}^{i,1}$, they can be accessed from $\mathbf{u_2}$ by $\mathbf{A_{\Gamma 2}}$ which also is the transpose of $\mathbf{A_{2\Gamma}}$.

Using the in- and outflow equations above in substituting the unknowns α_i and β_i in the Laplace equation, expressing each grid point in the interface yields a system of equations:

$$\begin{aligned}
u_{\mathbf{2}}^{1,n} - 4u_{\mathbf{\Gamma}}^{1} + u_{\mathbf{1}}^{1,1} + b_{\mathrm{N}}^{n+1} + u_{\mathbf{\Gamma}}^{2} &= 0 \\
u_{\mathbf{2}}^{2,n} - 4u_{\mathbf{\Gamma}}^{2} + u_{\mathbf{1}}^{2,1} + u_{\mathbf{\Gamma}}^{1} + u_{\mathbf{\Gamma}}^{3} &= 0 \\
u_{\mathbf{2}}^{3,n} - 4u_{\mathbf{\Gamma}}^{3} + u_{\mathbf{1}}^{3,1} + u_{\mathbf{\Gamma}}^{2} + u_{\mathbf{\Gamma}}^{4} &= 0 \\
&\vdots \\
u_{\mathbf{2}}^{n,n} - 4u_{\mathbf{\Gamma}}^{n} + u_{\mathbf{1}}^{n,1} + u_{\mathbf{\Gamma}}^{n-1} + b_{\mathrm{S}}^{n+1} &= 0
\end{aligned} \tag{2.12}$$

The coefficients acting on grid points in the interface can be expressed as a matrix in a similar fashion as for the subdomains according to Equation (2.11). Since there is one column of interface grid points, this is not a block matrix, but rather a sole matrix similar to the diagonal blocks in $\mathbf{A_{11}}$ and $\mathbf{A_{22}}$. This matrix is called $\mathbf{A_{\Gamma\Gamma}}$.

$$\underbrace{\left(\begin{array}{ccc|cc|c|cc}
\cdots & 0 & 1 & \cdots & 0 & & & \\
 & & 0 & & 1 & & & \vdots \\
 & & \vdots & & 0 & \ddots & & 0 \\
 & & & & \vdots & & \ldots 0 & 1
\end{array}\right)}_{\mathbf{A_{\Gamma 1}}} \mathbf{u_1} + \underbrace{\begin{pmatrix}
-4 & 1 & & 0 \\
1 & -4 & \ddots & \\
 & \ddots & \ddots & 1 \\
0 & & 1 & -4
\end{pmatrix}}_{\mathbf{A_{\Gamma\Gamma}}} \mathbf{u_\Gamma} +$$

$$+ \underbrace{\left(\begin{array}{ccc|cc|c|cc}
1 & 0 & \cdots & 0 & \cdots & & & \\
0 & & & 1 & & & \vdots & \\
\vdots & & & 0 & & \ddots & 0 & \\
 & & & \vdots & & & 1 & 0\ldots
\end{array}\right)}_{\mathbf{A_{\Gamma 2}}} \mathbf{u_2} = \underbrace{\begin{pmatrix}
-b_{\mathrm{N}}^{n+1} \\ 0 \\ \vdots \\ 0 \\ -b_{\mathrm{S}}^{n+1}
\end{pmatrix}}_{\mathbf{b_\Gamma}}$$

2.4 Summary of global problem formulation

The Laplace equation over each subdomains cannot be solved separately from each other, because the systems of linear equations contain unknowns from the interface, and the interface cannot be solved because it contains unknowns from both subdomains. However, by composing a large system of linear equations with all three sets of equations from each subdomain and interface, that system can be solved.

$$\begin{pmatrix} \mathbf{A_{11}} & \mathbf{A_{1\Gamma}} & \mathbf{0} \\ \mathbf{A_{\Gamma 1}} & \mathbf{A_{\Gamma\Gamma}} & \mathbf{A_{\Gamma 2}} \\ \mathbf{0} & \mathbf{A_{2\Gamma}} & \mathbf{A_{22}} \end{pmatrix} \begin{pmatrix} \mathbf{u_1} \\ \mathbf{u_\Gamma} \\ \mathbf{u_2} \end{pmatrix} = \begin{pmatrix} \mathbf{b_1} \\ \mathbf{b_\Gamma} \\ \mathbf{b_2} \end{pmatrix}. \tag{2.13}$$

These block matrices are visualised in Figure 2.9 to illustrate their respective sizes.

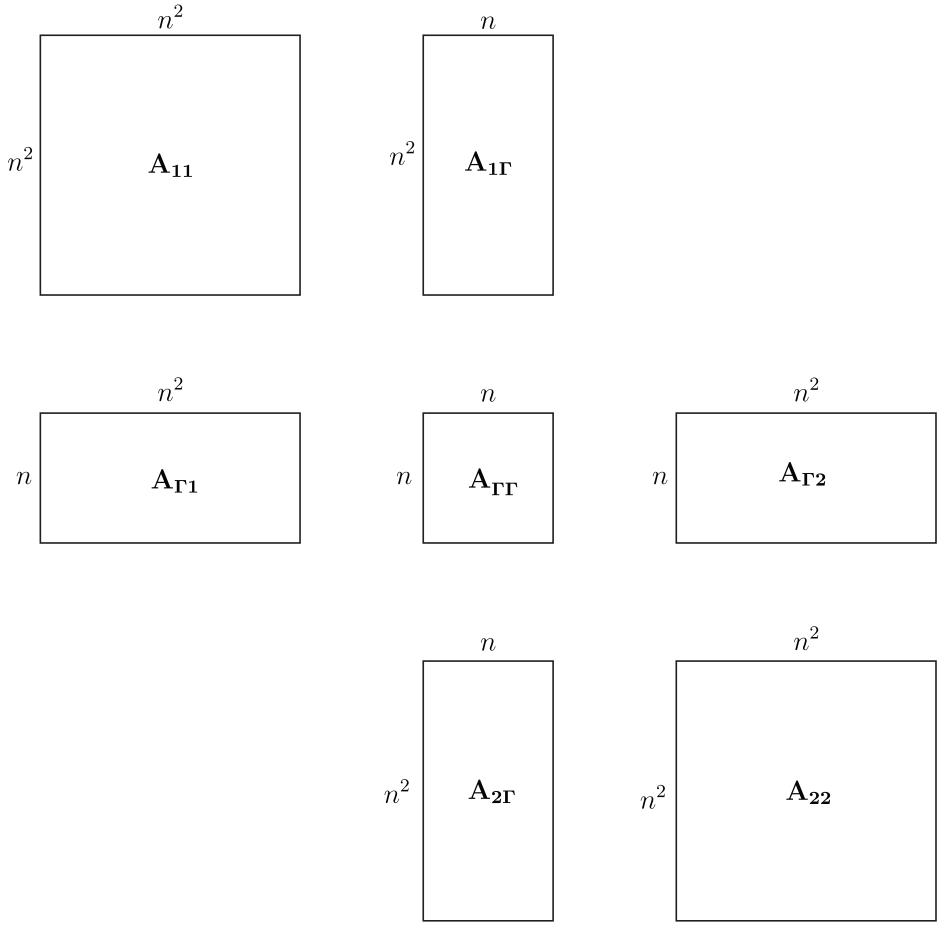


Figure 2.9: A graphical schematic of the different block matrices.

These matrices can be used as tools to formulate both Dirichlet-, and Neumann conditions based on how they are arranged.

Chapter 3

Subdomains-solving Problem Formulations

Instead of solving the Laplace equation over the entire room at once, it is possible to solve each subdomain separately. For this, it is necessary to formulate boundary conditions over each subdomain such that they can be solved independently of each other. The most basic method to achieve this is with the Dirichlet-Neumann algorithm, which works as follows:

By setting a Dirichlet boundary condition for the first subdomain, the system of linear equations for that subdomain can be solved with the interface as the values along the boundary. By setting a Neumann condition over the other subdomain, this will include the interface due to the discretisation of the derivative. By using the newly calculated first subdomain, the derivative over the interface is known and can be set as the boundary condition, and the system of linear equations for the second subdomain can be solved. In doing so, the interface will be calculated anew.

3.1 Dirichlet-Neumann algorithm

Starting with Ω_1, by setting a Dirichlet condition over the its boundary adjacent to the interface, the system of linear equations that arise from the discretisation can be solved. The boundary condition is that the values along the boundary should be the same as along the interface. These values are given by ∂_Γ.

For Ω_2 we impose a Neumann condition, that inflow and the outflow to be equal.

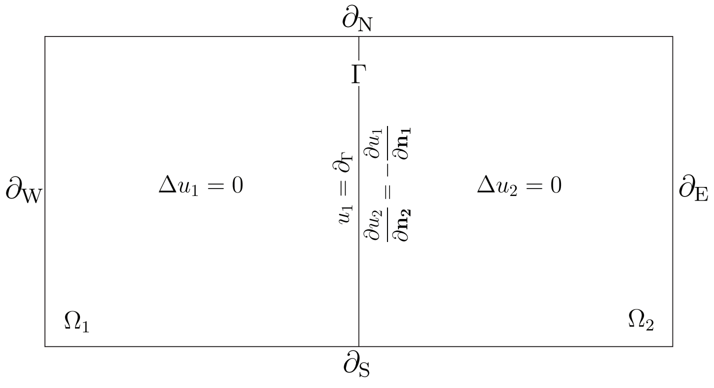


Figure 3.1: The domain with Dirichlet- and Neumann conditions over the interface.

The conditions that are now imposed over each subdomain are the following:

$$\Omega_1 : \begin{cases} \Delta u_1 = 0 & \text{in } \Omega_1 & i) & \text{The Laplace equation is applied to the internal points in } \Omega_1. \\ u_1 = g & \text{on } \partial_{\Omega_1} \setminus \Gamma & ii) & \text{The values along the boundary } \partial_{\Omega_1} \setminus \Gamma \text{ are given by } g. \\ u_1 = \partial_\Gamma & \text{on } \Gamma & iii) & \text{A Dirichlet condition is imposed along the interface, specifying the values on } u_1 \text{ along } \Gamma \text{ to have values given by } \partial_\Gamma. \end{cases} \tag{3.1}$$

$$\Omega_2 : \begin{cases} \Delta u_2 = 0 & \text{in } \Omega_2 & i) & \text{The Laplace equation is applied to the internal points in } \Omega_2. \\ u_2 = g & \text{on } \partial_{\Omega_2} \setminus \Gamma & ii) & \text{The values along the boundary } \partial_{\Omega_2} \setminus \Gamma \text{ are given by } g. \\ \frac{\partial u_1}{\partial \mathbf{n_1}} = -\frac{\partial u_2}{\partial \mathbf{n_2}} & \text{on } \Gamma & iii) & \text{A Neumann condition is imposed, specifying the derivatives for } \Omega_2 \text{ along the interface to be equal to those from } \Omega_1. \end{cases} \tag{3.2}$$

3.1.1 Imposing a Dirichlet condition on a subdomain

To impose a Dirichlet condition on Ω_1, the discretisation process here is almost the same as the one described in the previous chapter, but with a small difference. In order to solve the system of linear equations alone for Ω_1, the values for the Dirichlet condition along the boundary adjacent to the interface must be known. Assuming that the values over the interface are known, the system of linear equations can be solved. The asterisk indicates that the values are known.

$$\mathbf{A_{11}u_1} = \mathbf{b_1} - \mathbf{A_{1\Gamma}u_\Gamma}^*. \tag{3.3}$$

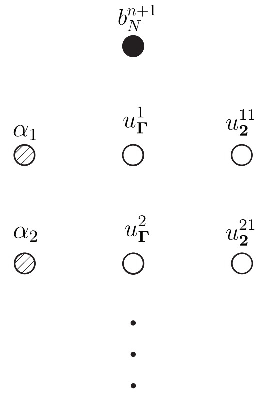
3.1.2 Imposing a Neumann condition on a subdomain

When imposing a Neumann condition on a subdomain, in this case Ω_2, it is done in two parts as the grid points along the interface now also are unknown. One condition is set up for the internal points, and another for the interface.

For the internal points, the discretisation process is the same as previously explained in **2.2 Finding the Approximate Temperature Distribution**, with the difference that the grid points along the interface remain unknown,

$$\mathbf{A_{2\Gamma}u_\Gamma} + \mathbf{A_{22}u_2} = \mathbf{b_2}. \tag{3.4}$$

For the grid points along the interface, the discretisation process is very similar to that when discretising the interface, except only one side has unknown values.



The inflow to Ω_2 is given by

$$-\frac{\partial u_2}{\partial \mathbf{n_2}} = -\frac{\alpha_i - u_{\mathbf{\Gamma}}^i}{\Delta x}$$

which has to be equal to the outflow from Ω_1 which is calculated from the known values from Ω_1 according to

$$\frac{\partial u_1}{\partial \mathbf{n_1}} = \frac{{u_{\mathbf{\Gamma}}^i}^* - {u_{\mathbf{1}}^{i,n}}^*}{\Delta x},$$

${u_{\mathbf{\Gamma}}^i}^*$ signifies the previously known discrete values from the interface, and ${u_{\mathbf{1}}^{i,n}}^*$ are the values from Ω_1 adjacent to the interface.

As the inflow and the outflow should be the same, α_i can be expressed in terms of knowns and unknowns of the subdomains:

$$\frac{{u_{\mathbf{\Gamma}}^i}^* - {u_{\mathbf{1}}^{i,n}}^*}{\Delta x} = -\frac{\alpha_i - u_{\mathbf{\Gamma}}^i}{\Delta x}$$

which can be rearranged to

$$\alpha_i = \underbrace{u_{\mathbf{\Gamma}}^i}_{\text{unknown}} \underbrace{- {u_{\mathbf{\Gamma}}^i}^* + {u_{\mathbf{1}}^{i,n}}^*}_{\text{known}}. \tag{3.5}$$

Discretising the Laplace equation over the interface is done as detailed in (2.11). In this case, the equations have the form

$$\alpha_i - 4u_{\mathbf{\Gamma}}^i + u_{\mathbf{2}}^{i1} + u_{\mathbf{\Gamma}}^{i+1} + u_{\mathbf{\Gamma}}^{i-1} = 0$$

Inserting α_i as expressed in (3.5) for all the equations that arise from the discretisation over the interface, the following system of linear equations is found

$$\begin{aligned}
-3u_{\mathbf{\Gamma}}^1 + u_{\mathbf{\Gamma}}^2 + \qquad\quad +u_{\mathbf{2}}^{11} &= -b_{\mathrm{N}}^{n+1} + {u_{\mathbf{\Gamma}}^1}^* - {u_{\mathbf{1}}^{1,n}}^* \\
u_{\mathbf{\Gamma}}^1 - 3u_{\mathbf{\Gamma}}^2 + u_{\mathbf{\Gamma}}^3 + \quad +u_{\mathbf{2}}^{21} &= \qquad\quad +{u_{\mathbf{\Gamma}}^2}^* - {u_{\mathbf{1}}^{2,n}}^* \\
+u_{\mathbf{\Gamma}}^2 - 3u_{\mathbf{\Gamma}}^3 + u_{\mathbf{\Gamma}}^4 + u_{\mathbf{2}}^{31} &= \qquad\quad +{u_{\mathbf{\Gamma}}^3}^* - {u_{\mathbf{1}}^{3,n}}^* \\
&\vdots \\
\underbrace{u_{\mathbf{\Gamma}}^{n-1} - 3u_{\mathbf{\Gamma}}^n}_{\mathbf{A_{\Gamma\Gamma}^{(II)} u_\Gamma}} \quad \underbrace{+u_{\mathbf{2}}^{n1}}_{\mathbf{A_{\Gamma 2} u_2}} &= \underbrace{-b_{\mathrm{S}}^{n+1}}_{\mathbf{b_\Gamma}} \underbrace{+{u_{\mathbf{\Gamma}}^n}^*}_{\mathbf{u_\Gamma^*}} \underbrace{-{u_{\mathbf{1}}^{n,n}}^*}_{\mathbf{A_{\Gamma 1} u_1}}.
\end{aligned} \tag{3.6}$$

As indicated by the brackets below the equations, this system of linear equations can be expressed by using the earlier introduced block matrices and a new $n \times n$ matrix $\mathbf{A_{\Gamma\Gamma}^{(II)}}$

$$\mathbf{A_{\Gamma\Gamma}^{(II)}} = \begin{pmatrix} -3 & 1 & & 0 \\ 1 & -3 & \ddots & \\ & \ddots & \ddots & 1 \\ 0 & & 1 & -3 \end{pmatrix}$$

The index $^{(\mathrm{II})}$ indicates that this matrix adheres to the second subdomain, Ω_2. With these matrices, the system of linear equations can be summarised as

$$\mathbf{A_{\Gamma\Gamma}^{(II)} u_\Gamma} + \mathbf{A_{\Gamma 2} u_2} = \mathbf{b_\Gamma} + \mathbf{u_\Gamma^*} - \mathbf{A_{\Gamma 1} u_1^*} \tag{3.7}$$

Summary of the Dirichlet-Neumann algorithm

Finding the temperature distribution over the room using the Dirichlet-Neumann algorithm entails solving each set of systems of linear equations

$$\Omega_1 : \Big\{ \mathbf{A_{11} u_1} = \mathbf{b_1} - \mathbf{A_{1\Gamma} u_\Gamma}^* \tag{3.8a}$$

$$\Omega_2 : \begin{cases} \mathbf{A_{\Gamma\Gamma}^{(II)} u_\Gamma} + \mathbf{A_{\Gamma 2} u_2} &= \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^* - \mathbf{A_{\Gamma 1} u_1}^* \\ \mathbf{A_{2\Gamma} u_\Gamma} + \mathbf{A_{22} u_2} &= \mathbf{b_2} \end{cases} \tag{3.8b}$$

These systems of linear equations introduced so far can be viewed as constraining the sets of unknowns to certain conditions. Solving the following systems of equations

$$\mathbf{A_{11}u_1} = \mathbf{b_1} - \mathbf{A_{1\Gamma}u_\Gamma}^*$$

can in words be expressed as finding the temperature distribution within Ω_1 with a Dirichlet condition over the adjacent interface Γ. In essence, the unknowns have to fulfill a set of critera imposed by the equations.

Having a set of unknowns fulfill the constraints imposed by the systems of linear equations in

$$\mathbf{A_{\Gamma\Gamma}^{(I)}u_\Gamma} + \mathbf{A_{\Gamma 1}u_1} = \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^* - \mathbf{A_{\Gamma 2}u_2}^* \tag{3.9}$$

can in words be put as imposing a Neumann condition for the interface and adjacent subdomain Ω_1. This only affects the interface and the sets of grid points immediately adjacent to the interface, and not the entire set of internal points in Ω_1. In order to solve the system of linear equations, the condition to fulfill the Laplace Equation has to be imposed over the internal points also. In combination with the interface grid points being unknown, $\mathbf{u_\Gamma}$ remains unknown in the system of linear equations for the internal points. This means that this set of equations in Equation (3.9) is not enough a constraint to find a temperature distribution over Ω_1 with a Neumann condition. These equations are the governing equations touching the interface and subdomain where the Neumann condition acts.

Imposing a Dirichlet condition on a subdomain:

$$\Omega_1 : \quad \mathbf{A_{11}u_1} = \mathbf{b_1} - \mathbf{A_{1\Gamma}u_\Gamma}^* \tag{3.10}$$

$$\Omega_2 : \quad \mathbf{A_{22}u_2} = \mathbf{b_2} - \mathbf{A_{2\Gamma}u_\Gamma}^* \tag{3.11}$$

Imposing a Neumann condition on a subdomain:

$$\Omega_1 : \quad \mathbf{A_{\Gamma\Gamma}^{(I)}u_\Gamma} + \mathbf{A_{\Gamma 1}u_1} = \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^* - \mathbf{A_{\Gamma 2}u_2}^* \tag{3.12}$$

$$\Omega_2 : \quad \mathbf{A_{\Gamma\Gamma}^{(II)}u_\Gamma} + \mathbf{A_{\Gamma 2}u_2} = \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^* - \mathbf{A_{\Gamma 1}u_1}^* \tag{3.13}$$

These systems of linear equations can in short be used as tools to impose constraints that act as Dirichlet- and Neumann-conditions over a given subdomain.

3.2 A proposition for a new approach

As touched upon in the introduction, the main theme of this thesis is to investigate if it is possible to find the temperature distribution when posing more than one boundary condition over each subdomain. This is done by setting up a system of linear equations containing the conditions for imposing a Dirichlet-, and a Neumann condition for each subdomain. The problem is then the following, as illustrated in Figure 3.2.

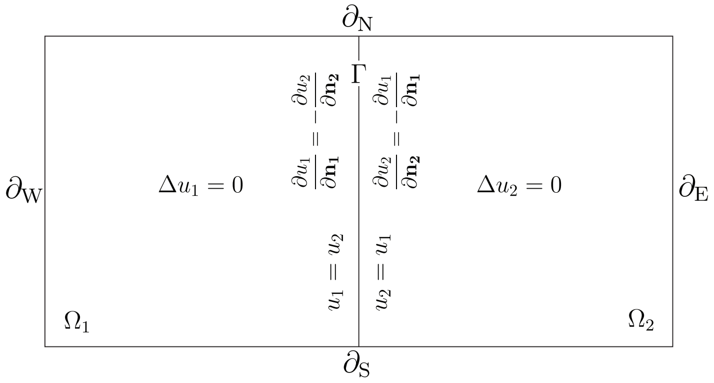


Figure 3.2: The subdomains with both Dirichlet- and Neumann conditions along the interface for each sudomain.

For each subdomain, there is a set of conditions that we want to impose:

For $\mathbf{\Omega_1}$:		**For $\mathbf{\Omega_2}$** :			
$\Delta u_1 = 0$	in Ω_1	$\Delta u_2 = 0$	in Ω_2	*i)*	Imposing the Laplace equation on the internal points over each subdomain.
$u_1 = g$	on $\partial_{\Omega_1} \setminus \Gamma$	$u_2 = g$	on $\partial_{\Omega_2} \setminus \Gamma$	*ii)*	Imposing the boundary values over each subdomain, not including the interface Γ.
$u_1 = u_2$	on Γ	$u_2 = u_1$	on Γ	*iii)*	Imposing the condition that the values on each subdomain must be continuous over Γ.
$\frac{\partial u_1}{\partial \mathbf{n_1}} = -\frac{\partial u_2}{\partial \mathbf{n_2}}$	on Γ	$\frac{\partial u_2}{\partial \mathbf{n_2}} = -\frac{\partial u_1}{\partial \mathbf{n_1}}$	on Γ	*iv)*	Imposing the condition that the inflow and outflow over Γ should be the same.

(3.14)

Setting up the conditions

By using the equations given in (3.10) - (3.13), it is possible to impose the conditions described in (3.14). These conditions set up the following sets of linear equations,

$$\Omega_1: \quad \mathbf{A_{11}u_1} = \mathbf{b_1} - \mathbf{A_{1\Gamma}u_\Gamma}^*$$

$$\Omega_2: \quad \mathbf{A_{22}u_2} = \mathbf{b_2} - \mathbf{A_{2\Gamma}u_\Gamma}^*$$

Solving internal points with Dirichlet condition along interface, Dirichlet data are the values from the assumed to be known, $\mathbf{u_\Gamma}^*$.

which fulfills conditions $(3.14)_i$ and $(3.14)_{ii}$,

$$\Omega_1 \text{ and } \Omega_2: \quad \mathbf{u_\Gamma} = \mathbf{u_\Gamma}^*$$

Values on Γ should be equal to $\mathbf{u_\Gamma}^*$.

which fulfills conditions $(3.14)_{iii}$,

$$\Omega_1: \quad \mathbf{A_{\Gamma 1}u_1} + \mathbf{A_{\Gamma\Gamma}^{(I)}u_\Gamma} = \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^* - \mathbf{A_{\Gamma 2}u_2}^*$$

$$\Omega_2: \quad \mathbf{A_{\Gamma\Gamma}^{(II)}u_\Gamma} + \mathbf{A_{\Gamma 2}u_2} = \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^* - \mathbf{A_{\Gamma 1}u_1}^*$$

Derivatives along Γ should equal derivatives calculated over $\mathbf{u_\Gamma}$ from a known adjacent subdomain.

which fulfills conditions $(3.14)_{iv}$.

For each subdomain, the unknowns will be $\mathbf{u_1}$, $\mathbf{u_\Gamma}$ and $\mathbf{u_2}$, $\mathbf{u_\Gamma}$ respectively. Solving each subdomain will produce its own set of solutions for the unknowns along the interface. In order to keep the interface for each subdomain separate, they will be referred to as $\mathbf{u_{\Gamma(1)}}$ and $\mathbf{u_{\Gamma(2)}}$. This also means that when solving the systems for each subdomain, the interface $\mathbf{u}_{\mathbf{\Gamma}}^*$ is different for each subdomain according to

$$\mathbf{u}_{\mathbf{\Gamma}}^* = \begin{cases} \mathbf{u}_{\mathbf{\Gamma(2)}}^* & \text{when solving over } \Omega_1 \\ \mathbf{u}_{\mathbf{\Gamma(1)}}^* & \text{when solving over } \Omega_2 \end{cases} .$$

When moving to an overdetermined formulation of the iteration matrices, there is need for a new notation. This is done by adding the subindex ${}_{\mathcal{O}_i}$, where i refers to the subdomain. Each set of unknowns per subdomain can be grouped together in vectors as

$$\mathbf{u}_{\mathcal{O}_1} = \begin{pmatrix} \mathbf{u_1} \\ \mathbf{u}_{\mathbf{\Gamma(1)}} \end{pmatrix} \text{ and } \mathbf{u}_{\mathcal{O}_2} = \begin{pmatrix} \mathbf{u}_{\mathbf{\Gamma(2)}} \\ \mathbf{u_1} \end{pmatrix}.$$

Taking the systems of linear equations that enforces the criteria in (3.14) and using the vectors $\mathbf{u}_{\mathcal{O}_1}$ and $\mathbf{u}_{\mathcal{O}_2}$ the following system can be set up

$$\Omega_1 : \quad \begin{pmatrix} \mathbf{A_{11}} & 0 \\ 0 & \mathbf{I} \\ \mathbf{A_{\Gamma 1}} & \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{I})} \end{pmatrix} \begin{pmatrix} \mathbf{u_1} \\ \mathbf{u}_{\mathbf{\Gamma(1)}} \end{pmatrix} = \begin{pmatrix} \mathbf{b_1} \\ 0 \\ \mathbf{b}_{\mathbf{\Gamma}} \end{pmatrix} + \begin{pmatrix} -\mathbf{A_{1\Gamma}} \mathbf{u}_{\mathbf{\Gamma(2)}}{}^* \\ \mathbf{u}_{\mathbf{\Gamma(2)}}{}^* \\ \mathbf{u}_{\mathbf{\Gamma(2)}}{}^* - \mathbf{A_{\Gamma 2}} \mathbf{u_2}{}^* \end{pmatrix} \tag{3.15a}$$

$$\Omega_2 : \quad \begin{pmatrix} \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})} & \mathbf{A_{\Gamma 2}} \\ \mathbf{I} & 0 \\ 0 & \mathbf{A_{22}} \end{pmatrix} \begin{pmatrix} \mathbf{u}_{\mathbf{\Gamma(2)}} \\ \mathbf{u_2} \end{pmatrix} = \begin{pmatrix} \mathbf{b}_{\mathbf{\Gamma}} \\ 0 \\ \mathbf{b_2} \end{pmatrix} + \begin{pmatrix} \mathbf{u}_{\mathbf{\Gamma(1)}}{}^* - \mathbf{A_{\Gamma 1}} \mathbf{u_1}{}^* \\ \mathbf{u}_{\mathbf{\Gamma(1)}}{}^* \\ -\mathbf{A_{2\Gamma}} \mathbf{u}_{\mathbf{\Gamma(1)}}{}^* \end{pmatrix}, \tag{3.15b}$$

or,

$$\begin{aligned} \mathbf{A}_{\mathcal{O}_1} \mathbf{u}_{\mathcal{O}_1} &= \mathbf{b}_{\mathcal{O}_1} + \mathbf{l}_{\mathcal{O}_1} \\ \mathbf{A}_{\mathcal{O}_2} \mathbf{u}_{\mathcal{O}_2} &= \mathbf{b}_{\mathcal{O}_2} + \mathbf{l}_{\mathcal{O}_2} \end{aligned}$$

where

$$\mathbf{A}_{\mathcal{O}_1} = \begin{pmatrix} \mathbf{A_{11}} & 0 \\ 0 & \mathbf{I} \\ \mathbf{A_{\Gamma 1}} & \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{I})} \end{pmatrix}, \quad \mathbf{b}_{\mathcal{O}_1} = \begin{pmatrix} \mathbf{b_1} \\ 0 \\ \mathbf{b}_{\mathbf{\Gamma}} \end{pmatrix}, \quad \mathbf{l}_{\mathcal{O}_1} = \begin{pmatrix} -\mathbf{A_{1\Gamma}} \mathbf{u}_{\mathbf{\Gamma(2)}}{}^* \\ \mathbf{u}_{\mathbf{\Gamma(2)}}{}^* \\ \mathbf{u}_{\mathbf{\Gamma(2)}}{}^* - \mathbf{A_{\Gamma 2}} \mathbf{u_2}{}^* \end{pmatrix}$$

$$\mathbf{A}_{\mathcal{O}_2} = \begin{pmatrix} \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})} & \mathbf{A_{\Gamma 2}} \\ \mathbf{I} & 0 \\ 0 & \mathbf{A_{22}} \end{pmatrix}, \quad \mathbf{b}_{\mathcal{O}_2} = \begin{pmatrix} \mathbf{b}_{\mathbf{\Gamma}} \\ 0 \\ \mathbf{b_2} \end{pmatrix}, \quad \mathbf{l}_{\mathcal{O}_2} = \begin{pmatrix} \mathbf{u}_{\mathbf{\Gamma(1)}}{}^* - \mathbf{A_{\Gamma 1}} \mathbf{u_1}{}^* \\ \mathbf{u}_{\mathbf{\Gamma(1)}}{}^* \\ -\mathbf{A_{2\Gamma}} \mathbf{u}_{\mathbf{\Gamma(1)}}{}^* \end{pmatrix}.$$

Because this approach is a new one, it has yet to be named. From this point onward it will be referred to as the Dirichlet & Neumann - Dirichlet & Neumann algorithm, or the DNDN-algorithm for short.

3.3 From algorithm to method

Up to this point, both the Dirichlet-Neumann-algorithm, and the Dirichlet & Neumann - Dirichlet & Neumann algorithm have been expressed as a set of systems of linear equations for each subdomain. In practice, in order to find a good approximation for the temperature distribution of the entire room, using a subdomain method involves iterations between the subdomains. Given an initial guess, each subdomain is solved in turn, giving new information to the other subdomain. This new information enters the new subdomain as those indices denoted by an asterisk. Moving to an iterative process, these indices will be denoted by the iteration step instead.

Chapter 4

Subdomain Methods

This chapter details how the Dirichlet-Neumann method and the DNDN-method finds the temperature distribution for the room. The Dirichlet-Neumann method is included in detail since it will be used as a benchmark to validate the efficacy of the newly proposed DNDN-method. The DNDN-method involves solving an overdetermined system of linear equations. How this is done is described in detail in this chapter as well.

An iterative process

These methods are iterative, in that an intermediate solution is found for each subdomain in turn, changing the values over the interface where information is sent between the subdomains. The practice is that one iteration step involves solving two systems of linear equations for each subdomain.

When the iteration reaches termination, the two solutions produced for each subdomain will be an adequate approximation of the temperature distribution for the entire the room, granted that the method works. The termination criterion used is

$$||\mathbf{u_\Gamma}^{k+1} - \mathbf{u_\Gamma}^{k}||_2 < \tau,$$

that is, when the change of the values over the interface is smaller than a certain tolerance τ, the iteration is complete [2]

Between each iteration, there is a practice of using relaxation to speed up the methods since the use of relaxation decreases the number of iterations needed to reach termination [3] [4] . For the DNDN-method, this is not as straightforward as for the DN-method, so how the DNDN-method is relaxed will be a topic for discussion in this chapter.

4.1 Dirichlet-Neumann Method

Given an initial guess, each system of linear equations for each subdomain is solved, updating the values of $\mathbf{u_1}$, $\mathbf{u_\Gamma}$, and $\mathbf{u_2}$. This is repeated, solving the system for each subdomain until the termination criterion is met, and the solutions from each system can be combined to approximate the temperature distribution of the room. The method is outlined as steps:

The Dirichlet-Neumann method:

Set an initial guess for $\mathbf{u_1}^0$, $\mathbf{u_\Gamma}^0$, $\mathbf{u_2}^0$

1. Solving over Ω_1:

$$\mathbf{A_{11}u_1}^{k+1} = \mathbf{b_1} - \mathbf{A_{1\Gamma}u_\Gamma}^k$$

2. Solving over Ω_2:

$$\begin{pmatrix} \mathbf{A}^{(\mathrm{II})}_{\mathbf{\Gamma\Gamma}} & \mathbf{A_{\Gamma 2}} \\ \mathbf{A_{2\Gamma}} & \mathbf{A_{22}} \end{pmatrix} \begin{pmatrix} \mathbf{u_\Gamma} \\ \mathbf{u_2} \end{pmatrix}^{k+1} = \begin{pmatrix} \mathbf{b_\Gamma} \\ \mathbf{b_1} \end{pmatrix} + \begin{pmatrix} \mathbf{u_\Gamma}^k - \mathbf{A_{\Gamma 1}u_1}^{k+1} \\ 0 \end{pmatrix}$$

3. Relaxation: $\mathbf{u_\Gamma}^{k+1} := \theta \mathbf{u_\Gamma}^{k+1} + (1-\theta)\mathbf{u_\Gamma}^k$

At termination, $||\mathbf{u}_\mathbf{\Gamma}^{k+1} - \mathbf{u_\Gamma}^k||_2 < \tau$, the iterated solution is given by $\mathbf{u}_\mathbf{1}^{k+1}$, $\mathbf{u}_\mathbf{\Gamma}^{k+1}$, $\mathbf{u}_\mathbf{2}^{k+1}$.

Step 1-3 is repeated until termination criterion is met.

4.2 Dirichlet & Neumann - Dirichlet & Neumann method

For the proposed method, the iteration structure is similar to the DN-method. Each system of linear equations for each subdomain is solved in turn, and betwee solving these systems, relaxation can be performed. Relaxation has proved to increase the convergence rate of the DN-iteration [4, p.400], relaxation is included in the DNDN method also. Because this method provides a solution that includes gridpoints from each subdomain and the interface, how relaxation should be carried out is not as clear as in the DN-method. In the stencil detailing the DNDN method, steps 2 and 4 indicate where potential relaxation can be carried out.

The Dirichlet & Neumann - Dirichlet & Neumann method:

Set an initial guess $\mathbf{u_1}^0$, $\mathbf{u_\Gamma}^0$, $\mathbf{u_2}^0$

1. Solving over Ω_1:

$$\begin{pmatrix} \mathbf{A_{11}} & 0 \\ 0 & \mathbf{I} \\ \mathbf{A_{\Gamma 1}} & \mathbf{A}^{(\mathrm{I})}_{\mathbf{\Gamma\Gamma}} \end{pmatrix} \begin{pmatrix} \mathbf{u_1} \\ \mathbf{u_{\Gamma(1)}} \end{pmatrix}^{k+1} = \begin{pmatrix} \mathbf{b_1} \\ 0 \\ \mathbf{b_\Gamma} \end{pmatrix} + \begin{pmatrix} -\mathbf{A_{1\Gamma}u_{\Gamma(2)}}^k \\ \mathbf{u_{\Gamma(2)}}^k \\ \mathbf{u_{\Gamma(2)}}^k - \mathbf{A_{\Gamma 2}u_2}^k \end{pmatrix} \tag{4.1}$$

2. Optional relaxation

3. Solving over Ω_2:

$$\begin{pmatrix} \mathbf{A}^{(\mathrm{II})}_{\mathbf{\Gamma\Gamma}} & \mathbf{A_{\Gamma 2}} \\ \mathbf{I} & 0 \\ 0 & \mathbf{A_{22}} \end{pmatrix} \begin{pmatrix} \mathbf{u_{\Gamma(2)}} \\ \mathbf{u_2} \end{pmatrix}^{k+1} = \begin{pmatrix} \mathbf{b_\Gamma} \\ 0 \\ \mathbf{b_2} \end{pmatrix} + \begin{pmatrix} \mathbf{u_{\Gamma(1)}}^{k+1} - \mathbf{A_{\Gamma 1}u_1}^{k+1} \\ \mathbf{u_{\Gamma(1)}}^{k+1} \\ -\mathbf{A_{2\Gamma}u_{\Gamma(1)}}^{k+1} \end{pmatrix} \tag{4.2}$$

4. Optional relaxation

At termination, $||\mathbf{u_{\Gamma(2)}}^{k+1} - \mathbf{u_{\Gamma(2)}}^k||_2 < \tau$, the iterated solution is given by $\mathbf{u}_\mathbf{1}^{k+1}$, $\mathbf{u}_\mathbf{\Gamma(2)}^{k+1}$, $\mathbf{u}_\mathbf{2}^{k+1}$

Step 1-4 is repeated until termination criterion is met.

4.3 Solving an overdetermined system

The systems of linear equations for each subdomain in the DNDN-method are overdetermined. An overdetermined system has either no solution, one solution, or infinitely many solutions. Given that we are looking to find the temperature distribution of the room, it is safe to say that there does exist one solution since we know from the reference solution that there is one unique temperature distribution for the given boundary conditions along the walls.

There are different ways of finding a solution to an overdetermined system. Most of them involve finding an approximate solution. In order to make an informed choice as to what approximation method to use, it is pertinent to ask the question: "what does a solution to an overdetermined system mean?". The least squares method provides an answer to this — a solution is one that minimizes the residual evenly to the overdetermined problem. This can be achieved by using the normal equation [1, p.386].

Definition 4.3.1. *A solution x in a least squares sense for an overdetermined system satisfies*

$$\min_x ||Ax - b||^2.$$

The solution x is found from the normal equations

$$x = (A^\intercal A)^{-1} A^\intercal b \tag{4.3}$$

4.3.1 The normal equations

The normal equations for the problem at hand involves large block matrices as introduced in the previous chapters. Expanding the normal equations and restructuring them will be helpful in later parts, and when implementing, fewer matrix calculations needs to be computed.

The systems of linear equations solved for each subdomain are

$$\begin{aligned} \mathbf{A}_{\mathcal{O}_1}\mathbf{u}_{\mathcal{O}_1}^{k+1} &= \mathbf{b}_{\mathcal{O}_1} + \mathbf{l}_{\mathcal{O}_1}^{k} \\ \mathbf{A}_{\mathcal{O}_2}\mathbf{u}_{\mathcal{O}_2}^{k+1} &= \mathbf{b}_{\mathcal{O}_2} + \mathbf{l}_{\mathcal{O}_2}^{k+1} \end{aligned}$$

Expanding the normal equations for Ω_1

For Ω_1, the normal equations are

$$(\mathbf{A}_{\mathcal{O}_1}^\intercal \mathbf{A}_{\mathcal{O}_1})^{-1}\mathbf{A}_{\mathcal{O}_1}^\intercal(\mathbf{b}_{\mathcal{O}_1} + \mathbf{l}_{\mathcal{O}_2}^{k}).$$

There is not much to be done regarding $(\mathbf{A}_{\mathcal{O}_1}^\intercal \mathbf{A}_{\mathcal{O}_1})^{-1}$, but $\mathbf{A}_{\mathcal{O}_1}^\intercal(\mathbf{b}_{\mathcal{O}_1} + \mathbf{l}_{\mathcal{O}_2}^{k})$ can be expanded and reorganised. As a reminder, the matrices involved in solving the overdetermined cases are as follows,

$$\mathbf{A}_{\mathcal{O}_1} = \begin{pmatrix} \mathbf{A_{11}} & 0 \\ 0 & \mathbf{I} \\ \mathbf{A_{\Gamma 1}} & \mathbf{A_{\Gamma\Gamma}^{(I)}} \end{pmatrix}, \quad \mathbf{b}_{\mathcal{O}_1} = \begin{pmatrix} \mathbf{b_1} \\ 0 \\ \mathbf{b_\Gamma} \end{pmatrix}, \quad \mathbf{l}_{\mathcal{O}_1} = \begin{pmatrix} -\mathbf{A_{1\Gamma}}\mathbf{u_{\Gamma(2)}}^k \\ \mathbf{u_{\Gamma(2)}}^k \\ \mathbf{u_{\Gamma(2)}}^k - \mathbf{A_{\Gamma 2}}\mathbf{u_2}^k \end{pmatrix}.$$

Expanding $\mathbf{A}_{\mathcal{O}_1}^\intercal(\mathbf{b}_{\mathcal{O}_1} + \mathbf{l}_{\mathcal{O}_2}^{k})$ gives us

$$\begin{aligned} \mathbf{A}_{\mathcal{O}_1}^\intercal(\mathbf{b}_{\mathcal{O}_1} + \mathbf{l}_{\mathcal{O}_1}^{k}) &= \begin{pmatrix} \mathbf{A_{11}^\intercal} & 0 & \mathbf{A_{\Gamma 1}^\intercal} \\ 0 & \mathbf{I} & \mathbf{A_{\Gamma\Gamma}^{(I)}}^\intercal \end{pmatrix} \begin{pmatrix} \mathbf{b_1} - \mathbf{A_{1\Gamma}}\mathbf{u_{\Gamma(2)}}^k \\ \mathbf{u_{\Gamma(2)}}^k \\ \mathbf{b_\Gamma} - \mathbf{A_{\Gamma 2}}\mathbf{u_2}^k + \mathbf{u_{\Gamma(2)}}^k \end{pmatrix} = \\ &= \begin{pmatrix} \mathbf{A_{11}^\intercal}\mathbf{b_1} - \mathbf{A_{11}^\intercal}\mathbf{A_{1\Gamma}}\mathbf{u_{\Gamma(2)}}^k + \mathbf{A_{\Gamma 1}^\intercal}\mathbf{b_\Gamma} - \mathbf{A_{\Gamma 1}^\intercal}\mathbf{A_{\Gamma 2}}\mathbf{u_2}^k + \mathbf{A_{\Gamma 1}^\intercal}\mathbf{u_{\Gamma(2)}}^k \\ \mathbf{u_{\Gamma(2)}}^k + \mathbf{A_{\Gamma\Gamma}^{(I)}}^\intercal\mathbf{b_\Gamma} - \mathbf{A_{\Gamma\Gamma}^{(I)}}^\intercal\mathbf{A_{\Gamma 2}}\mathbf{u_2}^k + \mathbf{A_{\Gamma\Gamma}^{(I)}}^\intercal\mathbf{u_{\Gamma(2)}}^k \end{pmatrix}, \end{aligned}$$

which contains elements that are dependent on the iteration step k, and elements that are constant. Separating these into two parts, $\mathbf{c}_1$ for the constant elements, and $\mathbf{L}_1$ for the elements that are dependent on the step k, we have

$$\mathbf{A}_{\mathcal{O}_1}^\intercal(\mathbf{b}_{\mathcal{O}_1} + \mathbf{l}_{\mathcal{O}_2}^{k}) = \underbrace{\begin{pmatrix} \mathbf{A_{11}^\intercal} & \mathbf{A_{\Gamma 1}^\intercal} \\ 0 & \mathbf{A_{\Gamma\Gamma}^{(I)}}^\intercal \end{pmatrix} \begin{pmatrix} \mathbf{b_1} \\ \mathbf{b_\Gamma} \end{pmatrix}}_{\mathbf{c}_1} + \underbrace{\begin{pmatrix} -\mathbf{A_{11}^\intercal}\mathbf{A_{1\Gamma}} + \mathbf{A_{\Gamma 1}^\intercal} & -\mathbf{A_{\Gamma 1}^\intercal}\mathbf{A_{\Gamma 2}} \\ \mathbf{I} + \mathbf{A_{\Gamma\Gamma}^{(I)}}^\intercal & -\mathbf{A_{\Gamma\Gamma}^{(I)}}^\intercal\mathbf{A_{\Gamma 2}} \end{pmatrix}}_{\mathbf{L}_1} \begin{pmatrix} \mathbf{u_{\Gamma(2)}}^k \\ \mathbf{u_2}^k \end{pmatrix}$$

which when put into the normal equations gives the equation

$$(\mathbf{A}_{\mathcal{O}_1}^\intercal \mathbf{A}_{\mathcal{O}_1})\mathbf{u}_{\mathcal{O}_1}^{k+1} = \mathbf{c}_1 + \mathbf{L}_1\mathbf{u}_{\mathcal{O}_2}^{k} \tag{4.4}$$

Expanding the normal equations for Ω_2

Similarly for Ω_2 we expand $\mathbf{A}_{\mathcal{O}_2}^{\top}(\mathbf{b}_{\mathcal{O}_2} + \mathbf{l}_{\mathcal{O}_2}^{k+1})$ where

$$\mathbf{A}_{\mathcal{O}_2} = \begin{pmatrix} \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})} & \mathbf{A}_{\mathbf{\Gamma 2}} \\ \mathbf{I} & 0 \\ 0 & \mathbf{A}_{\mathbf{22}} \end{pmatrix}, \quad \mathbf{b}_{\mathcal{O}_2} = \begin{pmatrix} \mathbf{b}_{\mathbf{\Gamma}} \\ 0 \\ \mathbf{b}_{\mathbf{2}} \end{pmatrix}, \quad \mathbf{l}_{\mathcal{O}_2} = \begin{pmatrix} {\mathbf{u}_{\mathbf{\Gamma(1)}}}^{k+1} - \mathbf{A}_{\mathbf{\Gamma 1}}{\mathbf{u}_{\mathbf{1}}}^{k+1} \\ {\mathbf{u}_{\mathbf{\Gamma(1)}}}^{k+1} \\ -\mathbf{A}_{\mathbf{2\Gamma}}{\mathbf{u}_{\mathbf{\Gamma(1)}}}^{k+1} \end{pmatrix}.$$

After cleaning up the right-hand side, the system takes the form

$$\mathbf{A}_{\mathcal{O}_2}^{\top}(\mathbf{b}_{\mathcal{O}_2} + \mathbf{l}_{\mathcal{O}_2}^{k+1}) = \underbrace{\begin{pmatrix} {\mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})}}^{\top} & 0 \\ \mathbf{A}_{\mathbf{\Gamma 2}}^{\top} & \mathbf{A}_{\mathbf{22}}^{\top} \end{pmatrix} \begin{pmatrix} \mathbf{b}_{\mathbf{\Gamma}} \\ \mathbf{b}_{\mathbf{2}} \end{pmatrix}}_{\mathbf{c}_2} + \underbrace{\begin{pmatrix} -{\mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})}}^{\top}\mathbf{A}_{\mathbf{\Gamma 1}} & {\mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})}}^{\top} + \mathbf{I} \\ -\mathbf{A}_{\mathbf{\Gamma 2}}^{\top}\mathbf{A}_{\mathbf{\Gamma 1}} & -\mathbf{A}_{\mathbf{22}}^{\top}\mathbf{A}_{\mathbf{2\Gamma}} + \mathbf{A}_{\mathbf{\Gamma 2}}^{\top} \end{pmatrix}}_{\mathbf{L}_2} \begin{pmatrix} {\mathbf{u}_{\mathbf{1}}}^{k+1} \\ {\mathbf{u}_{\mathbf{\Gamma(1)}}}^{k+1} \end{pmatrix}$$

which can with the above-introduced notation be summarised for both systems

$$\begin{aligned} (\mathbf{A}_{\mathcal{O}_1}^{\top}\mathbf{A}_{\mathcal{O}_1})\mathbf{u}_{\mathcal{O}_1}^{k+1} &= \mathbf{c}_1 + \mathbf{L}_1\mathbf{u}_{\mathcal{O}_2}^{k} \quad \text{for } \Omega_1 \\ (\mathbf{A}_{\mathcal{O}_2}^{\top}\mathbf{A}_{\mathcal{O}_2})\mathbf{u}_{\mathcal{O}_2}^{k+1} &= \mathbf{c}_2 + \mathbf{L}_2\mathbf{u}_{\mathcal{O}_1}^{k+1} \quad \text{for } \Omega_2. \end{aligned} \tag{4.5}$$

4.4 Constraining the least squares method

A solution given by using the least squares method is one where the method aims to minimize the residual evenly over the system, with no regards to the underlying mathematical problem at hand. This can be remedied however, using an extended version of the least squares method, constraining the unknowns to completely fulfill one set of conditions, restricting the residuals to be zero. The question is whether constraining the system makes the overdetermined method ”better”.

Definition 4.4.1. *Constraining a least squares system is done by adding a set of constraints $Cx = d$ whilst minimising the residual,*

$$\begin{array}{rc} \textit{Minimize} & ||Ax - b||^2 \\ \textit{subject to} & Cx = d \end{array}$$

where C and d set up the constraints whose residual has to be zero.

The minimal solution x to the original problem is then given by

$$\begin{pmatrix} A^{\top}A & C^{\top} \\ C & 0 \end{pmatrix} \begin{pmatrix} x \\ z \end{pmatrix} = \begin{pmatrix} A^{\top}b \\ d \end{pmatrix}, \tag{4.6}$$

where z are the Lagrange multipliers.

4.4.1 Choice of constraints

The idea is to use the conditions already present in the problem formulation. These conditions were introduced in the previous chapter, and is presented here also.

Equations (3.14)**:**

For $\boldsymbol{\Omega_1}$:		**For $\boldsymbol{\Omega_2}$** :			
$\Delta u_1 = 0$	in Ω_1	$\Delta u_2 = 0$	in Ω_2	*i)*	Imposing the Laplace equation on the internal points over each subdomain.
$u_1 = g$	on $\partial_{\Omega_1} \setminus \Gamma$	$u_2 = g$	on $\partial_{\Omega_2} \setminus \Gamma$	*ii)*	Imposing the boundary values over each subdomain, not including the interface Γ.
$u_1 = u_2$	on Γ	$u_2 = u_1$	on Γ	*iii)*	Imposing the condition that the values on each subdomain must be continuous over Γ.
$\frac{\partial u_1}{\partial \mathbf{n_1}} = -\frac{\partial u_2}{\partial \mathbf{n_2}}$	on Γ	$\frac{\partial u_2}{\partial \mathbf{n_2}} = -\frac{\partial u_1}{\partial \mathbf{n_1}}$	on Γ	*iv)*	Imposing the condition that the inflow and outflow over Γ should be the same.

Not all these conditions will have any effect on the problem. For instance, setting a constraint over $(3.14)_{iii}$ would mean that the interface cannot change, meaning it would remain the same as the initial guess — $\mathbf{u_\Gamma^0}$, and the method would terminate immediately, not producing a correct approximation. Condition $(3.14)_{ii}$ is only related to the outer boundary, which would not affect the solution. The conditions that can be used to experiment with are conditions $(3.14)_i$ and $(3.14)_{iv}$.

4.4.2 Constraining the internal points condition

The first approach is to constrain the condition that formulates the discretised Laplace equation over the internal points. In practice, this means that the condition $(3.14)_i$,

$$\begin{aligned}
\Omega_1 : \quad & \mathbf{A_{11}u_1} = \mathbf{b_1} - \mathbf{A_{1\Gamma}}{\mathbf{u_\Gamma}}^* \\
\Omega_2 : \quad & \mathbf{A_{22}u_2} = \mathbf{b_2} - \mathbf{A_{2\Gamma}}{\mathbf{u_\Gamma}}^*,
\end{aligned}$$

must be fulfilled without producing a residual for the constrained unknowns. By setting the matrices C and d in Equation (4.6) to the following,

$$\begin{aligned}
\Omega_1 : \quad & C = \begin{bmatrix}\mathbf{A_{11}} & 0\end{bmatrix} & \qquad \Omega_2 : \quad & C = \begin{bmatrix}0 & \mathbf{A_{22}}\end{bmatrix} \\
& d = \begin{bmatrix}\mathbf{b_1} - \mathbf{A_{1\Gamma}u_\Gamma^*}\end{bmatrix} & & d = \begin{bmatrix}\mathbf{b_2} - \mathbf{A_{2\Gamma}u_\Gamma^*}\end{bmatrix}.
\end{aligned}$$

these constraints are imposed for the overdetermined system. The method is then changed to the following:

The Constrained Internal Least Squares method:

Set an initial guess $\mathbf{u_1}^0$, $\mathbf{u_\Gamma}^0$, $\mathbf{u_2}^0$

1. Solving over Ω_1:

$$\begin{pmatrix} \mathbf{A}_{\mathcal{O}_1}^\intercal \mathbf{A}_{\mathcal{O}_1} & \begin{bmatrix} \mathbf{A_{11}}^\intercal \\ 0 \end{bmatrix} \\ \begin{bmatrix}\mathbf{A_{11}} & 0\end{bmatrix} & 0 \end{pmatrix} \begin{pmatrix} \begin{bmatrix}\mathbf{u_1}\\ \mathbf{u_{\Gamma(1)}}\end{bmatrix} \\ z \end{pmatrix}^{k+1} = \begin{pmatrix} \mathbf{c}_1 + \mathbf{L}_1\mathbf{u}_{\mathcal{O}_2}^k \\ \mathbf{b_1} - \mathbf{A_{1\Gamma}}\mathbf{u_{\Gamma(2)}}^k \end{pmatrix} \tag{4.7}$$

2. Optional relaxation

3. Solving over Ω_2:

$$\begin{pmatrix} \mathbf{A}_{\mathcal{O}_2}^\intercal \mathbf{A}_{\mathcal{O}_2} & \begin{bmatrix} 0 \\ \mathbf{A_{22}}^\intercal \end{bmatrix} \\ \begin{bmatrix}0 & \mathbf{A_{22}}\end{bmatrix} & 0 \end{pmatrix} \begin{pmatrix} \begin{bmatrix}\mathbf{u_{\Gamma(2)}}\\ \mathbf{u_2}\end{bmatrix} \\ z \end{pmatrix}^{k+1} = \begin{pmatrix} \mathbf{c}_2 + \mathbf{L}_2\mathbf{u}_{\mathcal{O}_1}^{k+1} \\ \mathbf{b_2} - \mathbf{A_{2\Gamma}}\mathbf{u_{\Gamma(1)}}^{k+1} \end{pmatrix} \tag{4.8}$$

4. Optional relaxation

At termination, $||\mathbf{u_{\Gamma(2)}}^{k+1} - \mathbf{u_{\Gamma(2)}}^k||_2 < \tau$, the iterated solution is given by $\mathbf{u_1}^{k+1}$, $\mathbf{u}_{\mathbf{\Gamma(2)}}^{k+1}$, $\mathbf{u_2}^{k+1}$

This approach is henceforth refered to as "Constrained Internal Least Squares", or "CILQ".

4.4.3 Constraining the derivatives condition

The second approach is to constrain the condition that formulates the inflow and outflow over the interface to be the same. This is done by having the condition $(3.14)_{iv}$,

$$\begin{aligned} \Omega_1 : \quad & \mathbf{A_{\Gamma 1}u_1} + \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{I})}\mathbf{u_\Gamma} = \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^* - \mathbf{A_{\Gamma 2}u_2}^* \\ \Omega_2 : \quad & \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})}\mathbf{u_\Gamma} + \mathbf{A_{\Gamma 2}u_2} = \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^* - \mathbf{A_{\Gamma 1}u_1}^* \end{aligned},$$

fulfilled without producing a residual for those unknowns in the condition. By setting the matrices C and d in Equation (4.6) to the following,

$$\begin{aligned} \Omega_1 : \quad C &= \begin{bmatrix}\mathbf{A_{\Gamma 1}} & \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{I})}\end{bmatrix} & \qquad \Omega_2 : \quad C &= \begin{bmatrix}\mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})} & \mathbf{A_{\Gamma 2}}\end{bmatrix} \\ d &= \begin{bmatrix}\mathbf{b_\Gamma} + \mathbf{u}_\mathbf{\Gamma}^* - \mathbf{A_{\Gamma 2}}\mathbf{u}_\mathbf{2}^*\end{bmatrix} & d &= \begin{bmatrix}\mathbf{b_\Gamma} + \mathbf{u}_\mathbf{\Gamma}^* - \mathbf{A_{\Gamma 1}}\mathbf{u}_\mathbf{1}^*\end{bmatrix}. \end{aligned}$$

these constraints are imposed for the overdetermined system. The method is then changed to the following:

The Constrained Derivatives Least Squares method:

Set an initial guess $\mathbf{u_1}^0$, $\mathbf{u_\Gamma}^0$, $\mathbf{u_2}^0$

1. Solving over Ω_1:

$$\begin{pmatrix} \mathbf{A}_{\mathcal{O}_1}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_1} & \begin{bmatrix} \mathbf{A_{\Gamma 1}}^{\mathsf{T}} \\ \mathbf{A_{\Gamma\Gamma}^{(I)}}^{\mathsf{T}} \end{bmatrix} \\ \begin{bmatrix} \mathbf{A_{\Gamma 1}} & \mathbf{A_{\Gamma\Gamma}^{(I)}} \end{bmatrix} & 0 \end{pmatrix} \begin{pmatrix} \begin{bmatrix} \mathbf{u_1} \\ \mathbf{u_{\Gamma(1)}} \end{bmatrix} \\ z \end{pmatrix}^{k+1} = \begin{pmatrix} \mathbf{c}_1 + \mathbf{L}_1\mathbf{u}_{\mathcal{O}_2}^k \\ \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^{\mathbf{k}} - \mathbf{A_{\Gamma 2}}\mathbf{u_2}^k \end{pmatrix} \tag{4.9}$$

2. Optional relaxation

3. Solving over Ω_2:

$$\begin{pmatrix} \mathbf{A}_{\mathcal{O}_2}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_2} & \begin{bmatrix} \mathbf{A_{\Gamma\Gamma}^{(II)}}^{\mathsf{T}} \\ \mathbf{A_{\Gamma 2}}^{\mathsf{T}} \end{bmatrix} \\ \begin{bmatrix} \mathbf{A_{\Gamma\Gamma}^{(II)}} & \mathbf{A_{\Gamma 2}} \end{bmatrix} & 0 \end{pmatrix} \begin{pmatrix} \begin{bmatrix} \mathbf{u_{\Gamma(2)}} \\ \mathbf{u_2} \end{bmatrix} \\ z \end{pmatrix}^{k+1} = \begin{pmatrix} \mathbf{c}_2 + \mathbf{L}_2\mathbf{u}_{\mathcal{O}_1}^{k+1} \\ \mathbf{b_\Gamma} + \mathbf{u_\Gamma}^{k+1} - \mathbf{A_{\Gamma 1}}\mathbf{u_1}^{k+1} \end{pmatrix} \tag{4.10}$$

4. Optional relaxation

At termination, $||\mathbf{u_{\Gamma(2)}}^{k+1} - \mathbf{u_{\Gamma(2)}}^k||_2 < \tau$, the iterated solution is given by $\mathbf{u_1}^{k+1}$, $\mathbf{u_{\Gamma(2)}}^{k+1}$, $\mathbf{u_2}^{k+1}$

This approach is henceforth referred to as ”Constrained Derivatives Least Squares”, or ”CDLQ”.

4.5 Choice of relaxation

In the Dirichlet-Neumann method, information between the iterations is transferred by the interface grid points, $\mathbf{u_\Gamma}$. For the DNDN-method, information is transfered by the interface and the subdomains. Therefore, the approach to relax the information-carrying vectors can be done in different ways. In this thesis, five approaches have been tested out.

For all DNDN-methods, the following scheme remains the same:

1. Solving over Ω_1 : $\begin{cases} \text{in:} & \mathbf{u_2}^k, \quad \mathbf{u_{\Gamma(2)}}^k \\ \text{out:} & \mathbf{u_{\Gamma(1)}}^{k+1}, \quad \mathbf{u_1}^{k+1} \end{cases}$

2. Solving over Ω_2 : $\begin{cases} \text{in:} & \mathbf{u_{\Gamma(1)}}^{k+1}, \quad \mathbf{u_1}^{k+1} \\ \text{out:} & \mathbf{u_2}^{k+1}, \quad \mathbf{u_{\Gamma(2)}}^{k+1} \end{cases}$

End relaxation

Relaxing interface after last step each iteration. This is the standard approach to relaxation. This type of relaxation is henceforth referred to as ”end” relaxation.

End relaxation:

Set an initial guess

1. Solving over Ω_1

2. Solving over Ω_2

3. Relax interface: $\mathbf{u_{\Gamma(2)}}^{k+1} := \theta\mathbf{u_{\Gamma(2)}}^{k+1} + (1-\theta)\mathbf{u_{\Gamma(2)}}^k$

Middle relaxation

Relaxing interface after intermediate and last step each iteration. This typ of relaxation is henceforth referred to as "middle" relaxation.

```
Middle relaxation:

    Set an initial guess
 1. Solving over $\Omega_1$
 2. Relax interface: $\mathbf{u_{\Gamma(1)}}^{k+1} = \theta\mathbf{u_{\Gamma(1)}}^{k+1} + (1-\theta)\mathbf{u_{\Gamma(1)}}^{k}$
 3. Solving over $\Omega_2$
 4. Relax interface: $\mathbf{u_{\Gamma(2)}}^{k+1} := \theta\mathbf{u_{\Gamma(2)}}^{k+1} + (1-\theta)\mathbf{u_{\Gamma(2)}}^{k}$
```

Full relaxation

Relaxing interface and subdomain after last step each iteration. Since these overdetermined methods send information between iterations by the interface and one of the subdomains, it might be a good idea to relax this subdomain as well. This typ of relaxation will be referred to as "full" relaxation later on.

```
Full relaxation:

    Set an initial guess
 1. Solving over $\Omega_1$
 2. Solving over $\Omega_2$
 3. Relax interface and subdomain: $\mathbf{u}_{\mathcal{O}_2}^{k+1} := \theta\mathbf{u}_{\mathcal{O}_2}^{k+1} + (1-\theta)\mathbf{u}_{\mathcal{O}_2}^{k}$
```

Remark 4.5.1. *Information is transferred by $\mathbf{u}_{\mathcal{O}_2}$ between iterations because Ω_1 is solved first. If Ω_2 were to be solved first, naturally, information would be transferred by $\mathbf{u}_{\mathcal{O}_1}$, and it would be relaxed instead.*

Accelerating the flux

Similar to relaxation, it is also possible to accelerate the flux, meaning multiplying the flux part of $\mathbf{l}_{\mathcal{O}_i}$ with θ

$$\mathbf{l}_{\mathcal{O}_1} = \begin{pmatrix} -\mathbf{A_{1\Gamma}u_{\Gamma(2)}}^{k} \\ \mathbf{u_{\Gamma(2)}}^{k} \\ \theta(\mathbf{u_{\Gamma(2)}}^{k} - \mathbf{A_{\Gamma 2}u_2}^{k}) \end{pmatrix}, \quad \mathbf{l}_{\mathcal{O}_2} = \begin{pmatrix} \theta(\mathbf{u_{\Gamma(1)}}^{k} - \mathbf{A_{\Gamma 1}u_1}^{k}) \\ \mathbf{u_{\Gamma(1)}}^{k} \\ -\mathbf{A_{2\Gamma}u_{\Gamma(1)}}^{k} \end{pmatrix}$$

Accelerating the load

It is also possible to accelerate the entire interface load, $\mathbf{l}_{\mathcal{O}_i}$ by scaling it by θ, as in $\theta\mathbf{l}_{\mathcal{O}_i}$.

4.6 The spectral radius of the iteration matrix

To further investigate the methods, here follows a deduction of the iteration matrices for each method. This is done to compare the spectral radius and the convergence rate of the methods. The iteration matrices differ based on how the method solves the overdetermined systems. The matrices for each system are very large, and it is therefore very difficult to carry out an analytical deduction of the eigenvalues. However, it is possible to find an approximation of the eigenvalues.

The largest eigenvalue of an iteration matrix forms the spectral radius [4] (p.394),

$$\rho(\Sigma) = \max_{\lambda\in\sigma(\Sigma)} |\lambda|$$

where Σ is the iteration matrix, the matrix that expresses the system of linear equations solving for both subdomains as one system. When $\rho(\Sigma) < 1$, the method is convergent.

4.6.1 Iteration matrices for Dirichlet & Neumann - Dirichlet & Neumann iteration

Standard least squares method

To find the iteration matrix for the overdetermined system, we begin by looking at the linear systems for each subdomain

$$(\mathbf{A}_{\mathcal{O}_1}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_1})\begin{pmatrix}\mathbf{u_1}\\ \mathbf{u_{\Gamma(1)}}\end{pmatrix}^{k+1} = \overbrace{\begin{pmatrix}\mathbf{A}_{\mathbf{11}}^{\mathsf{T}} & \mathbf{A}_{\mathbf{\Gamma 1}}^{\mathsf{T}}\\ 0 & \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{I})^{\mathsf{T}}}\end{pmatrix}\begin{pmatrix}\mathbf{b_1}\\ \mathbf{b_\Gamma}\end{pmatrix}}^{\mathbf{c}_1} + \overbrace{\begin{pmatrix}-\mathbf{A}_{\mathbf{11}}^{\mathsf{T}}\mathbf{A_{1\Gamma}} + \mathbf{A}_{\mathbf{\Gamma 1}}^{\mathsf{T}} & -\mathbf{A}_{\mathbf{\Gamma 1}}^{\mathsf{T}}\mathbf{A_{\Gamma 2}}\\ \mathbf{I} + \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{I})^{\mathsf{T}}} & -\mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{I})^{\mathsf{T}}}\mathbf{A_{\Gamma 2}}\end{pmatrix}}^{\mathbf{L}_1}\begin{pmatrix}\mathbf{u_{\Gamma(2)}}^{k}\\ \mathbf{u_2}^{k}\end{pmatrix}$$

$$(\mathbf{A}_{\mathcal{O}_2}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_2})\begin{pmatrix}\mathbf{u_{\Gamma(2)}}\\ \mathbf{u_2}\end{pmatrix}^{k+1} = \underbrace{\begin{pmatrix}\mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})^{\mathsf{T}}} & 0\\ \mathbf{A}_{\mathbf{\Gamma 2}}^{\mathsf{T}} & \mathbf{A}_{\mathbf{22}}^{\mathsf{T}}\end{pmatrix}\begin{pmatrix}\mathbf{b_\Gamma}\\ \mathbf{b_2}\end{pmatrix}}_{\mathbf{c}_2} + \underbrace{\begin{pmatrix}-\mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})^{\mathsf{T}}}\mathbf{A_{\Gamma 1}} & \mathbf{A}_{\mathbf{\Gamma\Gamma}}^{(\mathrm{II})^{\mathsf{T}}} + \mathbf{I}\\ -\mathbf{A}_{\mathbf{\Gamma 2}}^{\mathsf{T}}\mathbf{A_{\Gamma 1}} & -\mathbf{A}_{\mathbf{22}}^{\mathsf{T}}\mathbf{A_{2\Gamma}} + \mathbf{A}_{\mathbf{\Gamma 2}}^{\mathsf{T}}\end{pmatrix}}_{\mathbf{L}_2}\begin{pmatrix}\mathbf{u_1}^{k+1}\\ \mathbf{u_{\Gamma(1)}}^{k+1}\end{pmatrix}$$

By eliminating $\mathbf{u}_{\mathcal{O}_1}$ from $(\mathbf{A}_{\mathcal{O}_2}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_2})\mathbf{u}_{\mathcal{O}_2}^{k+1}$, we can express $\mathbf{u}_{\mathcal{O}_2}^{k+1}$ as depending on $\mathbf{u}_{\mathcal{O}_2}^{k}$ only. This system contains the iteration matrix Σ;

$$(\mathbf{A}_{\mathcal{O}_2}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_2})\mathbf{u}_{\mathcal{O}_2}^{k+1} = \mathbf{c}_2 + \mathbf{L}_2((\mathbf{A}_{\mathcal{O}_1}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_1})^{-1}\mathbf{c}_1 + (\mathbf{A}_{\mathcal{O}_1}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_1})^{-1}\mathbf{L}_1\mathbf{u}_{\mathcal{O}_2}^{k}).$$

Grouping together the constants as χ we have

$$\underbrace{(\mathbf{A}_{\mathcal{O}_2}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_2})}_{\Sigma_2}\mathbf{u}_{\mathcal{O}_2}^{k+1} = \chi + \underbrace{\mathbf{L}_2(\mathbf{A}_{\mathcal{O}_1}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_1})^{-1}\mathbf{L}_1}_{\Sigma_1}\mathbf{u}_{\mathcal{O}_2}^{k}.$$

After relaxation, $\mathbf{u}_{\mathcal{O}_2}^{k+1} := \theta\mathbf{u}_{\mathcal{O}_2}^{k+1} + (1-\theta)\mathbf{u}_{\mathcal{O}_2}^{k}$, the system of linear equations is expressed as

$$\mathbf{u}_{\mathcal{O}_2}^{k+1} = \theta(\Sigma_2^{-1}\chi + \Sigma_2^{-1}\Sigma_1\mathbf{u}_{\mathcal{O}_2}^{k}) + (1-\theta)\mathbf{u}_{\mathcal{O}_2}^{k}.$$

Reorganising further, putting all constants in χ', the spectral radius $\rho(\Sigma)$ can be found from

$$\mathbf{u}_{\mathcal{O}_2}^{k+1} = \theta\chi' + \underbrace{(I + \theta(\Sigma_2^{-1}\Sigma_1 - I))}_{\Sigma}\mathbf{u}_{\mathcal{O}_2}^{k}.$$

Constrained least squares methods

Finding the iteration matrix for a constrained least squares method involves finding the inverse of a very large matrix, and as such requires some tricks. The constrained system has the form

$$\begin{pmatrix}\mathbf{A}_{\mathcal{O}_i}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_i} & C_i^{\mathsf{T}}\\ C_i & 0\end{pmatrix}\begin{pmatrix}\mathbf{u}_{\mathcal{O}_i}\\ z\end{pmatrix} = \begin{pmatrix}\mathbf{A}_{\mathcal{O}_i}^{\mathsf{T}}(\mathbf{b}_{\mathcal{O}_i} + \mathbf{l}_{\mathcal{O}_i})\\ d_i\end{pmatrix},$$

where C_i and d_i depend on the constraint. Based on what method or which subdomain is used, the matrices change. However, regardless of what method is used, the matrices $\mathbf{A}_{\mathcal{O}_i}$ is always the same. Using the form for $\mathbf{A}_{\mathcal{O}_i}^{\mathsf{T}}(\mathbf{b}_{\mathcal{O}_i} + \mathbf{l}_{\mathcal{O}_i})$ given in (4.5), and taking advantage of the fact that d_i will in most cases have one constant element c_i' and one matrix $\mathbf{d}_i$ depending on which subdomain is evaluated. By separating the constant terms, we find the following expression,

$$\begin{pmatrix}A_{\mathcal{O}_i}^{\mathsf{T}}(\mathbf{b}_{\mathcal{O}_i} + \mathbf{l}_{\mathcal{O}_i})\\ d_i\end{pmatrix} = \begin{pmatrix}\mathbf{c}_1\\ c_1'\end{pmatrix} + \begin{pmatrix}\mathbf{L}_i\\ \mathbf{d}_i\end{pmatrix}\mathbf{u}_{\mathcal{O}_{i-1}}^{k}.$$

The constrained least squares methods can then be expressed as

$$\begin{pmatrix}\mathbf{A}_{\mathcal{O}_i}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_i} & C_i^{\mathsf{T}}\\ C_i & 0\end{pmatrix}\begin{pmatrix}\mathbf{u}_{\mathcal{O}_i}^{k+1}\\ z\end{pmatrix} = \begin{pmatrix}\mathbf{c}_1\\ c_1'\end{pmatrix} + \begin{pmatrix}\mathbf{L}_i\\ \mathbf{d}_i\end{pmatrix}\mathbf{u}_{\mathcal{O}_{i-1}}^{k}.$$

By identifying $\mathbf{d}_i$ for each constrained least squares method, it is possible to find the spectral radius, since constants do not affect the eigenvalues of an iteration matrix.

$\mathbf{A}_{\mathcal{O}_i}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_i}$ is always square and non-singular, which means that the inverse of the matrix can be found using the formula for block matrix inversion. For simplicity, the following notation will be used

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{B}(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1} & -\mathbf{A}^{-1}\mathbf{B}(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1} \\ -(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1} & (\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1} \end{pmatrix} := \begin{pmatrix} \mathbf{A}^\star & \mathbf{B}^\star \\ \mathbf{C}^\star & \mathbf{D}^\star \end{pmatrix}$$

Using the formula for block matrix inversion, we have that

$$\begin{pmatrix} \mathbf{A}_{\mathcal{O}_i}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_i} & C_i^{\mathsf{T}} \\ C_i & 0 \end{pmatrix}^{-1} = \begin{pmatrix} \mathbf{A}^\star & \mathbf{B}^\star \\ \mathbf{C}^\star & \mathbf{D}^\star \end{pmatrix}, \tag{4.11}$$

and with this inverse, it is possible to find the iteration matrix Σ for a constrained least squares method.

Since the constants do not affect the eigenvalues of the iteration matrix, they can be disregarded throughout the process. Taking advantage of the block inversion, the systems of linear equations for each subdomains can be expressed as

$$\Omega_1: \begin{pmatrix} \mathbf{u_1} \\ \mathbf{u_{\Gamma(1)}} \end{pmatrix}^{k+1} = c' + \begin{pmatrix} \mathbf{A_1}^\star & \mathbf{B_1}^\star \end{pmatrix} \begin{pmatrix} \mathbf{L}_1 \\ \mathbf{d}_1 \end{pmatrix} \mathbf{u}_{\mathcal{O}_2}^k$$

$$\Omega_2: \begin{pmatrix} \mathbf{u_{\Gamma(2)}} \\ \mathbf{u_2} \end{pmatrix}^{k+1} = c' + \begin{pmatrix} \mathbf{A_2}^\star & \mathbf{B_2}^\star \end{pmatrix} \begin{pmatrix} \mathbf{L}_2 \\ \mathbf{d}_2 \end{pmatrix} \mathbf{u}_{\mathcal{O}_1}^{k+1}$$

Next, we want to eliminate $\mathbf{u}_{\mathcal{O}_1}^{k+1}$ from the second system of linear equations, to express the iteration in terms of one system of linear equations, and therefore one iteration matrix. After grouping all constants together as χ, gives us

$$\begin{pmatrix} \mathbf{u_{\Gamma(2)}} \\ \mathbf{u_2} \end{pmatrix}^{k+1} = \chi + \underbrace{\begin{pmatrix} \mathbf{A_1}^\star & \mathbf{B_1}^\star \end{pmatrix} \begin{pmatrix} \mathbf{L}_1 \\ \mathbf{d}_1 \end{pmatrix} \begin{pmatrix} \mathbf{A_2}^\star & \mathbf{B_2}^\star \end{pmatrix} \begin{pmatrix} \mathbf{L}_2 \\ \mathbf{d}_2 \end{pmatrix}}_{\Sigma'} \mathbf{u}_{\mathcal{O}_2}^k \tag{4.12}$$

When including relaxation, this becomes

$$\mathbf{u}_{\mathcal{O}_2}^{k+1} = \theta\chi + \underbrace{(I + \theta(\Sigma' - I))}_{\Sigma} \mathbf{u}_{\mathcal{O}_2}^k \tag{4.13}$$

where Σ is called the iteration matrix. The only thing that changes when employing different constraints are the matrices $\mathbf{A}_i^\star$, $\mathbf{B}_i^\star$, and $\mathbf{d_i}$.

Iteration matrix for Constrained Internal Least Squares

For this constraint, the C and d-matrix in Equation (4.6) are

$$\Omega_1: \begin{aligned} C &= \begin{bmatrix} \mathbf{A_{11}} & 0 \end{bmatrix} \\ d &= \begin{bmatrix} \mathbf{b_1} - \mathbf{A_{1\Gamma}}\mathbf{u}_{\mathbf{\Gamma}}^* \end{bmatrix} \end{aligned} \qquad \Omega_2: \begin{aligned} C &= \begin{bmatrix} 0 & \mathbf{A_{22}} \end{bmatrix} \\ d &= \begin{bmatrix} \mathbf{b_2} - \mathbf{A_{2\Gamma}}\mathbf{u}_{\mathbf{\Gamma}}^* \end{bmatrix} \end{aligned}$$

Now, since d only depends on $\mathbf{u_\Gamma}$, we need to add a matrix in order to have it work on the entire vector $\mathbf{u}_{\mathcal{O}_i}$. This is simply done by adding a matrix of zeros, giving the $\mathbf{d}_i$ matrices the shape

$$\begin{aligned} \mathbf{d}_1^{CILQ} &= \begin{bmatrix} \mathbf{A_{1\Gamma}} & 0 \end{bmatrix} \\ \mathbf{d}_2^{CILQ} &= \begin{bmatrix} 0 & \mathbf{A_{2\Gamma}} \end{bmatrix} \end{aligned}$$

Tt is now possible to deduce the matrices $\mathbf{A}_i^\star$, $\mathbf{B}_i^\star$ which depend on C according to Equation (4.11), and the iteration matrix Σ by using $\mathbf{d_i}$ according to Equations (4.12) and (4.13).

Iteration matrix for Constrained Derivative Least Squares

For this constraint, the C and d-matrix in Equation (4.6) are

$$\begin{aligned} \Omega_1: \quad C &= \begin{bmatrix} \mathbf{A}_{\boldsymbol{\Gamma}\mathbf{1}} & \mathbf{A}^{(\mathrm{I})}_{\boldsymbol{\Gamma\Gamma}} \end{bmatrix} & \qquad \Omega_2: \quad C &= \begin{bmatrix} \mathbf{A}^{(\mathrm{II})}_{\boldsymbol{\Gamma\Gamma}} & \mathbf{A}_{\boldsymbol{\Gamma}\mathbf{2}} \end{bmatrix} \\ d &= \begin{bmatrix} \mathbf{b}_{\boldsymbol{\Gamma}} - (\mathbf{A}_{\boldsymbol{\Gamma}\mathbf{2}}\mathbf{u}_{\mathbf{2}}^{*} - \mathbf{u}_{\boldsymbol{\Gamma}}^{*}) \end{bmatrix} & d &= \begin{bmatrix} \mathbf{b}_{\boldsymbol{\Gamma}} + (\mathbf{u}_{\boldsymbol{\Gamma}}^{*} - \mathbf{A}_{\boldsymbol{\Gamma}\mathbf{1}}\mathbf{u}_{\mathbf{1}}^{*}) \end{bmatrix} \end{aligned}$$

The same trick is applied here, only now does d_i depend both on $\mathbf{u_i}$ and $\mathbf{u}_{\boldsymbol{\Gamma}}$. The $\mathbf{d}_i$ matrices are given by

$$\begin{aligned} \mathbf{d}_1^{CDLQ} &= \begin{bmatrix} I & -\mathbf{A}_{\boldsymbol{\Gamma}\mathbf{2}} \end{bmatrix} \\ \mathbf{d}_2^{CDLQ} &= \begin{bmatrix} -\mathbf{A}_{\boldsymbol{\Gamma}\mathbf{1}} & I \end{bmatrix} \end{aligned}$$

Having this identified, it is possible to deduce the matrices $\mathbf{A}_i^{\star}$, $\mathbf{B}_i^{\star}$ which depend on C according to Equation (4.11), and the iteration matrix Σ by using $\mathbf{d_i}$ according to Equations (4.12) and (4.13).

4.6.2 Summary of the iteration matrices

The spectral radius of the iteration matrices for the overdetermined methods are found from the iteration matrix Σ.

For the Standard Least Squares method, the iteration matrix Σ is given by

$$\mathbf{u}_{\mathcal{O}_2}^{k+1} = \theta\chi' + \underbrace{(I + \theta(\Sigma_2^{-1}\Sigma_1 - I))}_{\Sigma}\mathbf{u}_{\mathcal{O}_2}^{k} \quad \text{where} \quad \begin{aligned} \Sigma_2 =& (\mathbf{A}_{\mathcal{O}_2}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_2}) \\ \Sigma_1 =& \mathbf{L}_2(\mathbf{A}_{\mathcal{O}_1}^{\mathsf{T}}\mathbf{A}_{\mathcal{O}_1})^{-1}\mathbf{L}_1 \end{aligned} \tag{4.14}$$

For a constrained least squares method, the iteration matrix Σ is given by

$$\mathbf{u}_{\mathcal{O}_2}^{k+1} = \theta\chi + \underbrace{(I + \theta(\Sigma' - I))}_{\Sigma}\mathbf{u}_{\mathcal{O}_2}^{k} \quad \text{where} \quad \Sigma' = \begin{pmatrix} \mathbf{A_1}^{\star} & \mathbf{B_1}^{\star} \end{pmatrix} \begin{pmatrix} \mathbf{L}_1 \\ \mathbf{d}_1 \end{pmatrix} \begin{pmatrix} \mathbf{A_2}^{\star} & \mathbf{B_2}^{\star} \end{pmatrix} \begin{pmatrix} \mathbf{L}_2 \\ \mathbf{d}_2 \end{pmatrix} \tag{4.15}$$

where Σ is given by C and $\mathbf{d}_i$, depending on the constraints.

Chapter 5

Results

Validating the methods

In order to validate the approximated solution produced by the iterative methods, the error e^k at step k is given by the difference between the intermediate solution $\mathbf{u}^k$ and reference solution $\mathbf{u}^r$,

$$e^k = ||\mathbf{u}^k - \mathbf{u}^r||_2 \text{ where } \mathbf{u}^k = \begin{pmatrix} \mathbf{u_1}^k \\ \mathbf{u_\Gamma}^k \\ \mathbf{u_2}^k \end{pmatrix}.$$

The reference solution $\mathbf{u^r}$ is found by solving the Laplace Equation over the room as one domain with the same number of internal points.

The speed of a method can be evaluated looking at the rate of convergence μ, given by

$$\mu = \frac{|e^{k+1}|}{|e^k|}.$$

The Dirichlet-Neumann method is included as a comparison when evaluating the efficiency of the methods. It is chosen as it is both easy to understand and to implement, whilst also producing good results.

The same number of grid points have been used for the methods to avoid discrepancies when comparing methods, namely $n = 18$. The tolerance $\tau = 1e^{-12}$ has been used.

Firstly, all methods will be presented individually. This will be followed by a closer look at how the choice of relaxation affects the methods. After that, the observed convergene rate and spectral radius is presented. Lastly, a summary of the optimal cases for all methods is presented. Each method has its own color, and the relaxation type is illustrated with different types of lines in the figures.

Method, colour representation

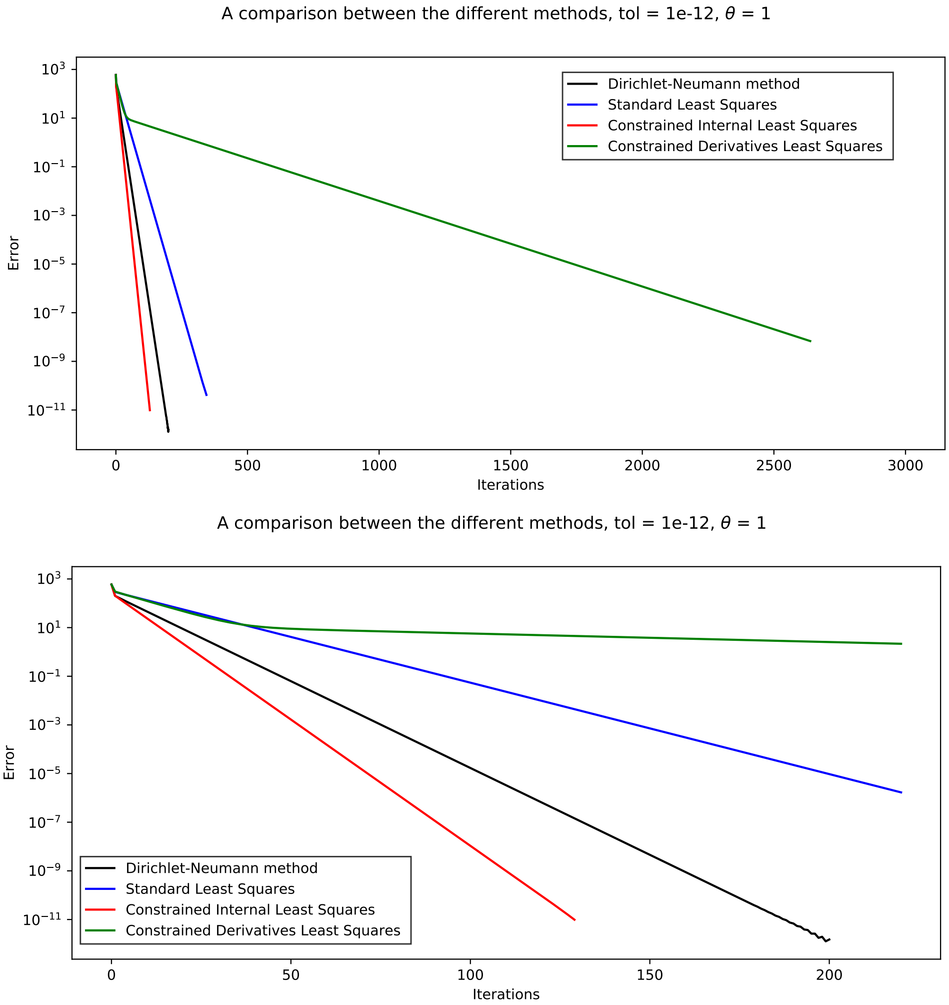
Dirichlet-Neumann method
Standard Least Squares
Constrained Internal Least Squares
Constrained Derivatives Least Squares

Relaxation, line representation

-⊖- End relaxation
—×— Full Relaxation
··●·· Middle relaxation

5.1 Methods

The following figures show two plots where the error decreases per iteration, without relaxation. This is done as to give a hint of the quality of the methods.


A comparison between the different methods, tol = 1e-12, θ = 1

A comparison between the different methods, tol = 1e-12, θ = 1

Figure 5.1: Error plot for the different methods, using no relaxation scheme.

The figures above show that the fastest method is the overdetermined method constraining the internal points condition. Next is presented how the relaxation modifies the performance of the methods.

5.2 Types of relaxation

As is evident from the figures, the type of relaxation plays a part in the rate of convergence. Here follows the three types of relaxation attempted, where the methods at termination found a good approximation of the temperature distribution.

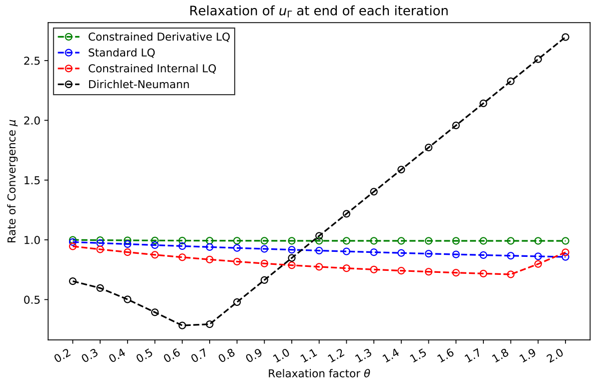


Figure 5.2: End relaxation

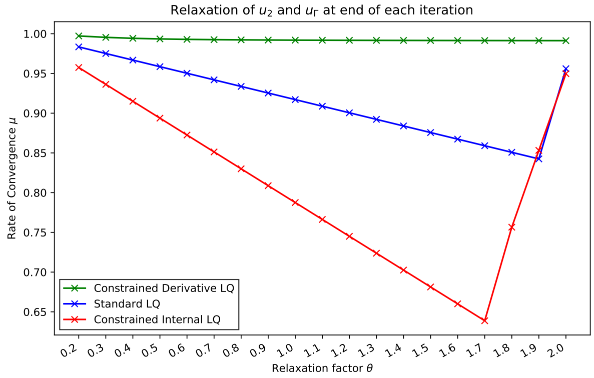


Figure 5.3: Full relaxation

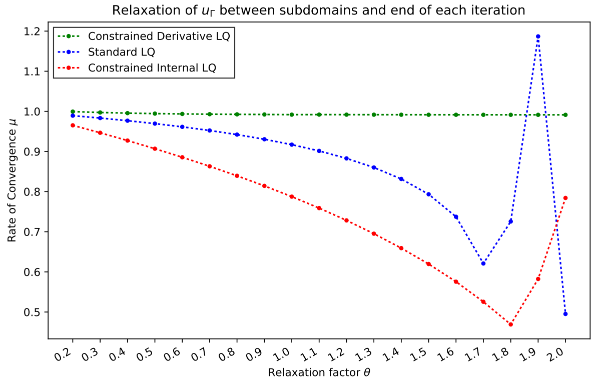


Figure 5.4: Middle relaxation

Once the relaxation factor θ is larger than 1, the Dirichlet-Neumann method diverges. By looking at the figures, it is apparent that Middle relaxation appears to be the best in terms of convergence rate. However, when looking at the error, see Figures 5.5 and 5.6, it is evident that the methods behaves strangely.

A closer look at full- and middle relaxation

When attempting the different relaxation types, the middle relaxation shows an odd behaviour which raises questions regarding if it is correct or not. The wavy way the error decreases over iterations hints to something being incorrect.

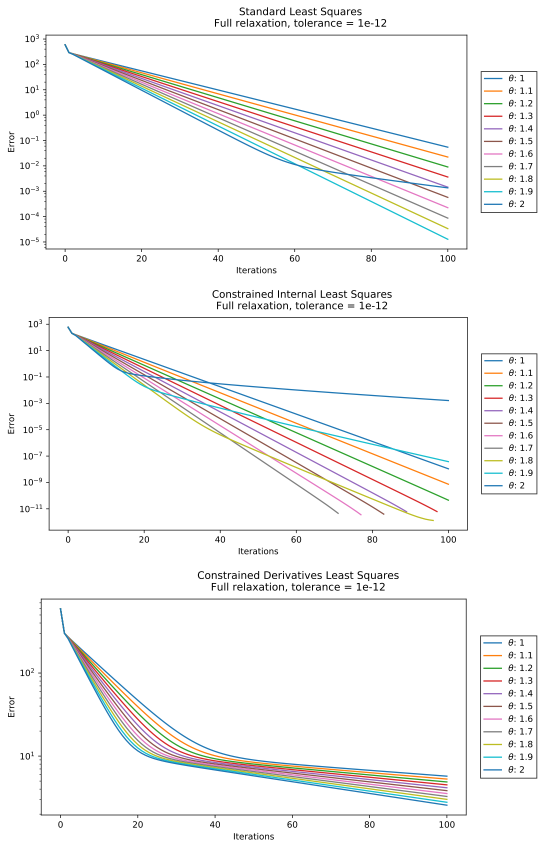


Figure 5.5: Error for all overdetermined methods using full relaxation.

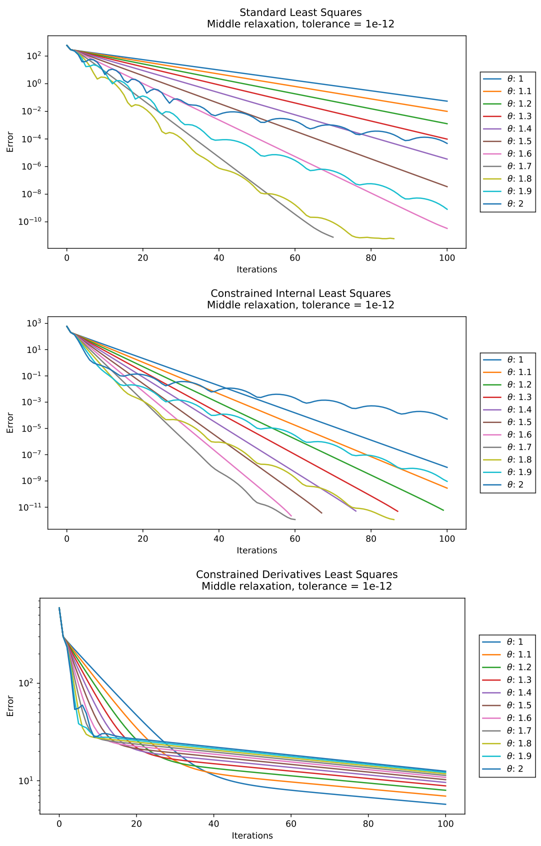


Figure 5.6: Error for all overdetermined methods using middle relaxation.

Pushing the relaxation factor

At first glance, it seems that a relaxation factor of $\theta = 2$ is the upper limit, but on closer inspection this turns out to be wrong. Pushing the relaxation factor to a value of 2.1 leads to some interesting results. For values higher than 2.1, all methods diverge. As can be seen in the figure below, both full- and middle relaxation become unstable, but the end relaxation still yields a good approximation of the temperature distribution, albeit very slowly compared to relaxation factors below 2.

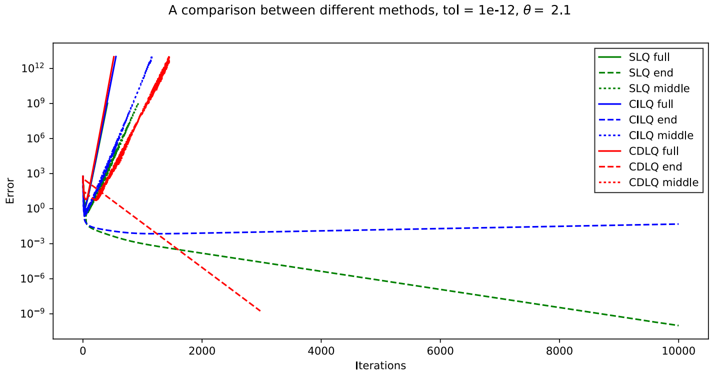


Figure 5.7: All methods and relaxation types pushing the relaxation factor to $\mu = 2.1$.

Flux- and load acceleration

Accelerating the flux and load does not result in a valid solution to the problem, as is apparent in the error plots in the figure below. These results are found using the CILQ method. Similar results are found for the other overdetermined methods also. Only for one value does the method yield a good result, for $\theta = 1$, i.e when there is no relaxation.

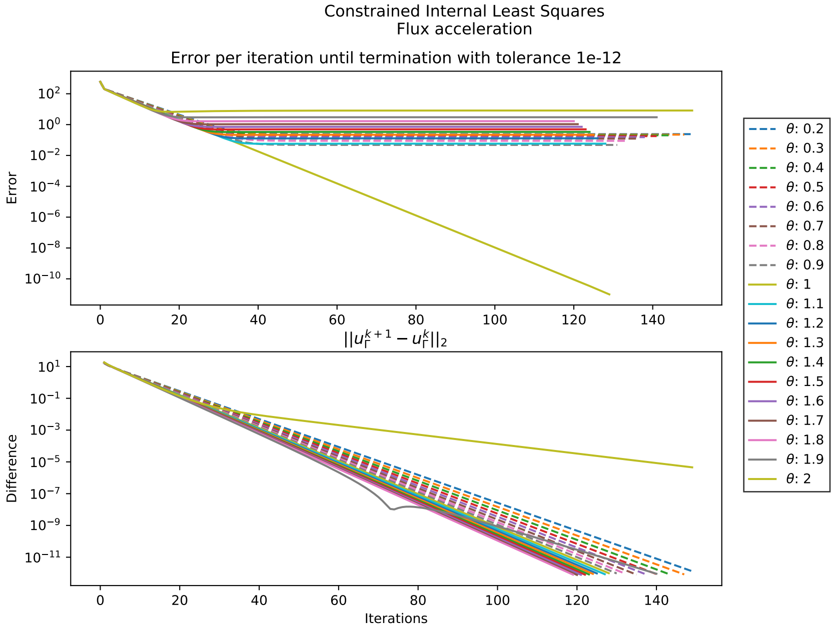


Figure 5.8: Constrained Internal Least Squares using flux acceleration relaxation.

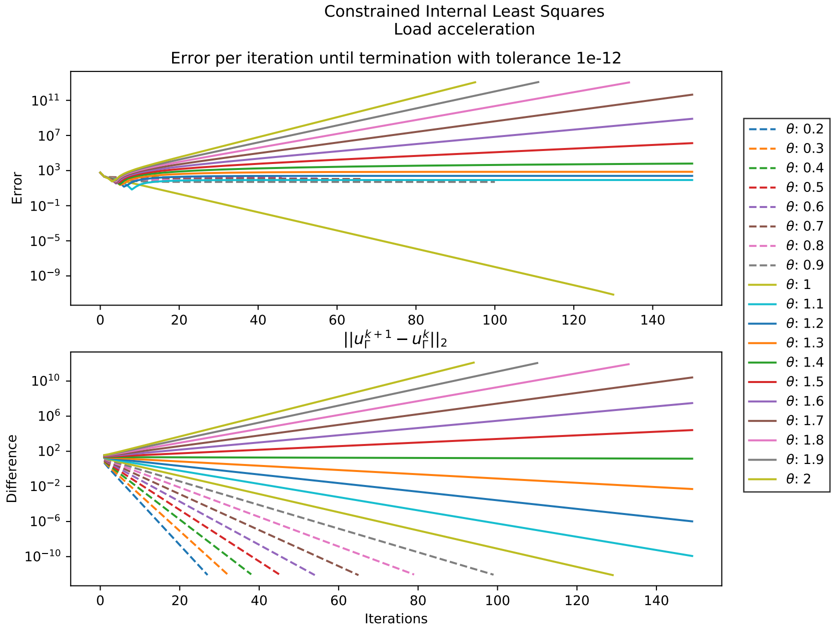


Figure 5.9: Constrained Internal Least Squares using load acceleration relaxation.

5.3 Convergence rate and spectral radius

Adding more grounds on which to discuss the methods, a comparison between the spectral radii of their respective iteration matrices and the observed convergence rate μ is shown in the figures below.

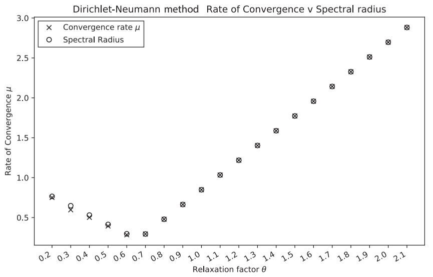


Figure 5.10: Convergence rate and spectral radius for the Dirichlet-Neumann method.

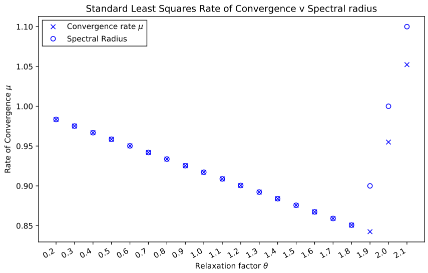


Figure 5.11: Convergence rate and spectral radius for the standard least squares method.

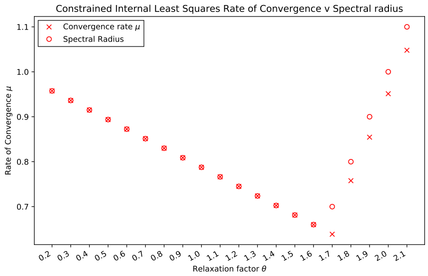


Figure 5.12: Convergence rate and spectral radius for the constrained internal least squares method.

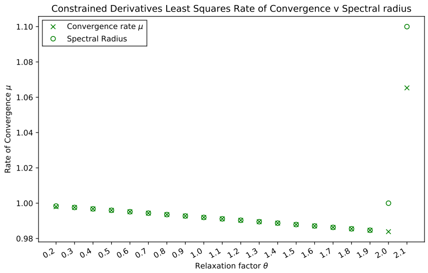


Figure 5.13: Convergence rate and spectral radius for the constrained derivative least squares method.

5.4 Summary

In order to properly validate the efficiency of the different approaches, all methods with their optimal relaxation and relaxation factor are presented. Values in parenthesis are the number of iterations required to reach termination for $\theta = 1$.

Table 5.1: A summary of the optimal conditions for each method.

Method	Relaxation Type	θ	Iterations	Convergence rate μ
Dirichlet-Neumann method	End	0.6	23 (199)	0.283421
Standard Least Squares	Full	1.9	180 (343)	0.842471
Constrained Internal Least Squares	Full	1.7	70 (128)	0.638589
Constrained Derivative Least Square	Full	2.0	1358 (2637)	0.983869

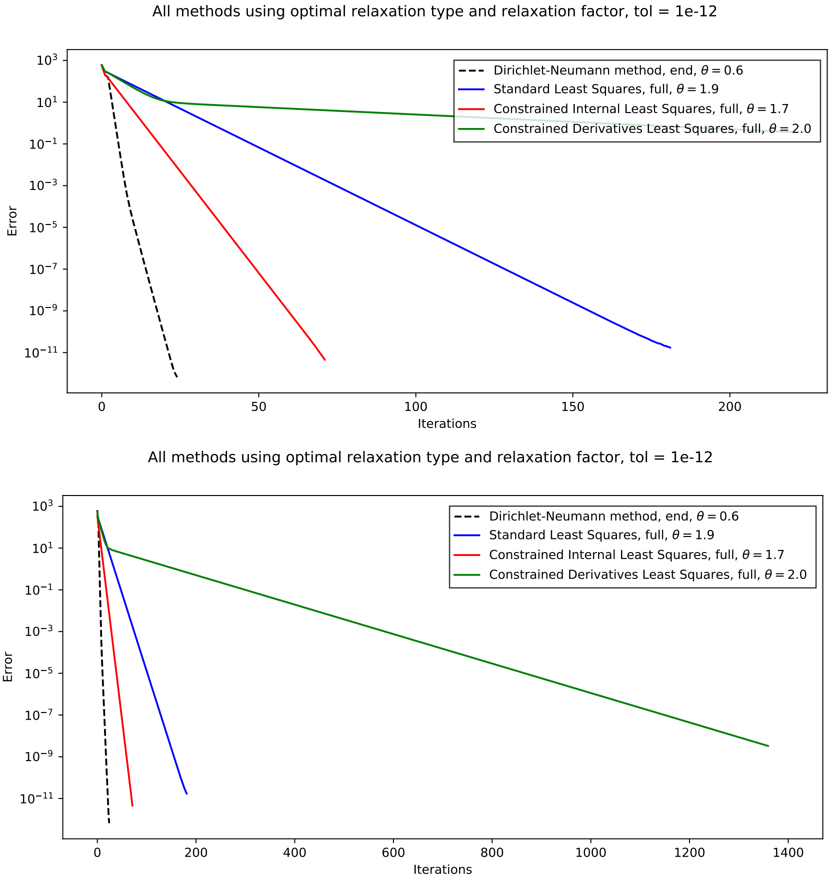


Figure 5.14: Error over iterations for all methods using the optimal relaxation type and relaxation factor.

From the table above, it is obvious that when looking at the optimal conditions for each method, the Dirichlet-Neumann method is the best in terms of number of iterations. This shows how much of a role the relaxation factor plays, as for the case where no relaxation is used, the overdetermined, constrained internal points method proved best.

Chapter 6

Discussion

6.1 Overdetermined domain decomposition methods

In general, the results provided show that overdetermined domain decomposition methods work. Figure 5.1 shows a comparison of all the different methods, without relaxation. Here it is clearly shown that there are significant differences in how well the different methods work. When not using relaxation, the constrained internal least squares method works the best, beating not only the other overdetermined methods, but also the standard Dirichlet Neumann method, by a few iterations.

6.2 Evaluation of methods

Looking at the different ways to constrain the least squares method, Figure 5.1 clearly shows that the choice of constraints grealy affect the speed of the method.

When starting off, the initial hypothesis was that constraining the derivatives condition would yield the best result. After some initial calculations, it became obvious that this is completely wrong. As Figure 5.1 shows, without relaxation, the method which reaches termination the quickest is the one where the internal points condition is constrained.

There has been no deeper research done as to why this is the case, though here follows a hypothesis. What influences the subdomains is the interface, since it is the only boundary that changes between iterations. If the interface is known, an adequate solution could be provided in one iteration. Since it is unknown, an initial guess has to be made. Then, for each iteration, the interface gets closer and closer to the "true" values, the values along the interface provided by the reference solution in this case. Therefore, what determines the amount of iterations required is how much the interface values can change for each iteration.

In the case where the derivatives condition has to be fulfilled without residual, the change is small between iterations, and therefore requires many iterations to reach termination. In the case where the internal points condition produce no residual, the condition to fulfill the derivatives condition leaves the residual there to be quite large, allowing larger changes in the interface for each iteration. For this reason, that method requires fewer iterations. In the case where no constraints are made, the residual is distributed evenly for all unknowns, which is why the standard least squares method is better than the CDLQ method, and worse than the CILQ method.

6.3 Evaluation of relaxation type

Figures 5.2, 5.3 and 5.4 show that the relaxation type greatly affects the efficiency of the methods. For the Dirichlet-Neumann method, the end relaxation is the only type that can be employed, and it greatly affects the number of iterations required to reach termination. Looking at Table 5.1, the difference is quire remarkable. The unrelaxed Dirichlet-Neumann method requires 199 iterations, whilst at the optimal relaxation at $\theta = 0.6$, only 23 iterations are required. The overdetermined methods work fairly well with this scheme as well, but their best results are found using middle relaxation. However, because the strange behaviour exhibited in Figure 5.6, comparing it to the full relaxation shown in Figure 5.5 which does not behave weirdly, there appears to be something wrong with the middle relaxation scheme.

After ruling out middle relaxation for the overdetermined methods, Figures 5.2 and 5.3 show that the most efficient relaxation is the full relaxation. For CILQ, the optimal relaxation of $\theta = 1.7$ reaches termination after 70 iterations as compared to 128 when not relaxed. For CDLQ, with $\theta = 2.0$, the number of iterations required is 1358 as compared to 2637 when not relaxed. For SLQ, with $\theta = 1.9$ the number of iterations required is 180 compared to 343.

One interesting observation is that the overdetermined methods remain stable for relaxation values over $\theta = 1$, compared to the Dirichlet-Neumann method which only reaches termination for values $\theta \leq 1$. For values larger than one, the relaxation factor acts as a form of accelerator for the overdetermined methods, and it is within the range $1 < \theta \leq 2$ the methods have their lowest convergence rate. $\theta > 2$ produced wrong solutions, but after closer investigation it seems that the all overdetermined methods with end relaxation eventually reaches termination with a value of $2 \leq \theta \leq 2.1$ as can be seen in Figure 5.7, albeit with many more iterations compared to values of $\theta \leq 2$.

The flux- and load relaxation appear not to work, in the sense that the solution at termination is not a valid approximation of the temperature distribution, as can be viewed in Figures 5.8 and 5.9.

Whilst initally the constrained internal least squares seemed the 'best' method, given the optimal value on θ the convergence rate μ is lowered greatly for the Dirichlet-Neumann method, surpassing all other methods in terms of number of iterations to termination.

6.4 Spectral radius

The spectral radii of the methods were investigated briefly. Figures 5.10, 5.11, 5.12, and 5.13 shows that for values of $0 < \theta < 1.6$ the spectral radius and the convergence rate match well. There are slight differences for lower values of θ for the Dirichlet-Neumann method, which is a result of the convergence rate values being difficult to measure since the method reaches termination very quickly. For values above $\theta = 1.6$, the convergence rates and spectral radii for the overdetermined methods stop coinciding. The CILQ method sees a difference for $\theta \geq 1.6$, SLQ for $\theta \geq 1.8$ and CDLQ for $\theta \geq 1.9$.

6.5 Conclusion

The goal was to answer the following questions;

- whether a domain decomposition method with more than one boundary condition over the interface can find a good approximation to the temperature distribution,
- how to handle the overdetermined systems of linear equations that arises when employing more than one boundary condition for each subdomain over the interface in a domain decomposition method,
- whether modifying the least squares method by constraining it using conditions found in the problem formulation work worse, as good as, or better than a basic domain decomposition method,
- how these modifications affect the methods.

Figure 5.1 shows that a domain decomposition method can be formulated with two boundary conditions over the interface, which will at termination have found a good approximation of the temperature distribution. These overdetermined methods includes the use of the the least squares method, or a constrained least squares method with different types of constraints. The choice of constraint greatly influence the convergence rate of the overdetermined methods. Using the condition to fulfill the Laplace Equation for the internal points proves to decrease the convergence rate, making it the best in terms of iterations, when comparing it to the method without relaxation.

Relaxation also proves to affect the methods. Several varieties have been attempted, where some work and some do not. Using the full relaxation scheme appears to be the best method, decreasing the number of iterations required to reach termination for all overdetermined methods. However, when comparing the overtermined methods with the standard Dirichlet-Neumann method using each methods optimal relaxation, the standard Dirichlet-Neumann method appears to be the best in terms of iterations required.

To summarise, the use of constrained least squares method seems to yield some interesting results, and should be investigated further. For optimal values, the Dirichlet-Neumann method proves most efficient.

Chapter 7

Future Work

Overdetermined domain decomposition methods remain fairly uninvestigated, and there are a great deal of topics to further research. What can be drawn from this thesis is that overdetermined methods does indeed find a good approximation to the temperature distribution at termination. In this thesis, the Laplace equation was investigated. The next step would be to investigate the Poisson equation.

For handling the overdetermined systems of linear equations, the use of a constrained least squares method was used. A number of constraints have been tested, where some 'worked' — meaning at termination, the solution produced is a good approximation to the problem, and some did not. There are more ways to mix the conditions and potentially setting up other constraints. In addition to constrained least squares methods, other approaches to solve the overdetermined systems of linear equations should be investigated, such as the use of weighted least squares method.

For all methods, a central difference discretisation was used. The most standard approach when dealing with domain decomposition methods is the use of the Finite Element method to solve for each subdomain. This could be an interesting topic to look into — both solving for the entire domain, and also when solving for one subdomain with a Finite Element method, and the other subdomain with a central difference scheme. Additionally, as touched upon in [2], large differences in material constants affects the speed of the solvers. It would be intersting to see if the same is true when employing an overdetermined subdomain formulation.

An initial step was taken in the investigation of the spectral radii, where the eigenvalues were approximated. Both looking at finding the true eigenvalues instead of an approximation, and further looking into what happens to the spectral radii for the methods when approaching $\theta = 2$, would be interesting.

In short, there are many interesting topics regarding the use of overdetermined domain decomposition methods, and the hope is that this thesis can be used as a stepping stone for these investigations.

Bibliography

[1] Nicholas J Higham. *Accuracy and stability of numerical algorithms.* SIAM, 2002.

[2] Azahar Monge and Philipp Birken. The Dirichlet-Neumann iteration for unsteady thermal fluid structure interaction. *Licentiate Thesis, Lund University*, 2016.

[3] Alfio Quarteroni and Alberto Valli. *Domain decomposition methods for partial differential equations.* Number CMCS-BOOK-2009-019. Oxford University Press, 1999.

[4] Andrea Toselli and Olof B Widlund. *Domain decomposition methods: algorithms and theory*, volume 34. Springer, 2005.