Delft University of Technology

Development of the Helmholtz Solver based on Schwarz Domain Decomposition Preconditioning

Erik Sieburgh

student number: 4611799

FACULTY OF ELECTRICAL ENGINEERING, MATHEMATICS AND COMPUTER SCIENCE

May 23, 2022

Contents

1 Introduction

The purpose of this chapter is to give an introduction to the work presented in this literature study, which deals with numerical solvers for the Helmholtz equation. The modeling of wave scattering in a efficient computational manner is crucial for many practical problems. The Helmholtz equation often arises from problems related to steady-state oscillations, such as electromagnetic, acoustical, mechanical and thermal problems. As can be seen in the following chapter, the Helmholtz equation follows from the wave equation by separation of variables and the Helmholtz equation represents a time-independent from of the wave equation.

Typically, finite difference or finite element methods are used for modeling of the Helmholtz equation. Additionally, in order for the bounded the Helmholtz problem to be well-posed Dirichlet boundary conditions or Sommerfeld radiation conditions are used. However, Dirichlet conditions cause wave reflections, while Sommerfeld conditions dampen these wave reflections.
As a consequence of either the finite difference or finite element modeling techniques a linear system of equations arises. For this linear system the coefficient matrix is large, sparse, symmetric and complex non-Hermitian. The coefficient matrix becomes indefinite very quickly when the wave number of the Helmholtz equation increases. Solving the indefinite Helmholtz equation numerically leads to two problems.

The first problem that arises is that when the wave number of the Helmholtz equation increases a pollution error appears in the solution. In order to avoid this pollution error the grid needs to be refined when the wave number increases. This gives rise to the second problem, which is that the problem size becomes very large for large wave numbers and that the problem size is dependent on the wave number.
Over the years a lot of active research has been devoted to developing numerical solvers for the Helmholtz problem that are wave number independent. This means that the wave number, and therefore the problem size, can increase, but the computational time stays the same. No numerical Helmholtz solver has yet been developed that has this independence property for real-world problems.

The goal of this report is to give an introduction into numerical Helmholtz solvers that deal with the wave number dependence and the large problem size of the Helmholtz problem. Additionally, the aim of this report is to give background information for the research questions that are proposed in Chapter 6. These research questions are the core of the thesis and research that will follow. The content of this report gives information necessary to construct and understand the research questions at the end.

In the first chapter after this introduction a proper introduction the the Helmholtz equation is given. This includes the derivation, the analytical solution and details about the boundary conditions used with the Helmholtz equation. In Chapter 3 details about the linear system, iterative methods and preconditioners for the Helmholtz problem are given. Additionally, existing relevant numerical Helmholtz solvers are mentioned and their details are given. Chapter 4 gives details about using domain decomposition techniques for preconditioning of iterative methods for the Helmholtz problem. After that an introduction to parallel computing is given in Chapter 5, since parallel computing can be used in combination with some domain decomposition techniques. The final chapter gives the details of two numerical experiments of test problems related to the work in this report. But more importantly, the final chapter states the research questions which are the core of the thesis that will follow.

2 | The Helmholtz Equation

This chapter is the start of the analysis of the Helmholtz equation. The Helmholtz equation is named after the physicist Hermann von Helmholtz who is the creator of the equation. It is a second order partial differential equation that models wave phenomena. The equation seems many applications in fields such as optics, acoustics, seismology and more.
The structure of this chapter is as follows: first the derivation of the Helmholtz equation from the homogeneous wave equation is given, followed by an explanation of different boundary conditions and finally, the construing of the analytical solution to the one dimensional non-homogeneous Helmholtz equation. This analytical solution is will be used thought out this study for analysis.

2.1 Derivation

The homogeneous wave equation is

$$\left(\frac{\partial^2}{\partial t^2} - c^2\nabla^2\right) u(\mathbf{x}, t) = 0, \tag{2.1}$$

where $u(\mathbf{x}, t)$ denotes the wave displacement.

Let us now apply the method of separation of variables. We begin by separating the solution $u(\mathbf{x}, t)$ in a time dependent and time independent part. This solution is then written in the following form:

$$u(\mathbf{x}, t) = h(t)\phi(\mathbf{x}). \tag{2.2}$$

Substituting (2.2) into (2.1) gives the results

$$\left(\frac{\partial^2}{\partial t^2} - c^2\nabla^2\right) h(t)\phi(\mathbf{x}) = 0. \tag{2.3}$$

By moving the time dependent parts to the left side, the space dependent parts to the right side and setting this equation equal the separation constant which we pick to be $-k^2$, we get the following equation

$$\frac{1}{c^2 h(t)}\frac{d^2 h(t)}{dt^2} = \frac{1}{\phi(\mathbf{x})}\nabla^2\phi(\mathbf{x}) = -k^2. \tag{2.4}$$

Thus we obtain two equations

- $$\frac{d^2 h(t)}{dt^2} = -k^2 c^2 h(t), \tag{2.5}$$
- $$\nabla^2\phi(\mathbf{x}) = -k^2\phi(\mathbf{x}). \tag{2.6}$$

Now, (2.6) gives is the Helmholtz equation, which can be rewritten to the more conventional form of

$$\left(-\nabla^2 - k^2\right)\phi(\mathbf{x}) = 0. \tag{2.7}$$

with k being the wave number, which is defined as

$$k = \frac{2\pi}{\lambda}, \tag{2.8}$$

where λ is the wave length.

The non-homogeneous Helmholtz equation can be written when the right hand side of (2.7) is replaced by a function $f(\mathbf{x})$. For this literature study, the non-homogeneous part is defined as the source function given by

$$f(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_0). \tag{2.9}$$

The non-homogeneous Helmholtz equation is therefore given by

$$\left(-\nabla^2 - k^2\right)\phi(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_0). \tag{2.10}$$

2.2 Boundary Conditions

In order to solve the Helmholtz equation on a domain Ω, boundary conditions are necessary for the problem to not be ill-posed. In problems with the Helmholtz equation the following boundary conditions are generally used. Also note that for a domain Ω the boundary of that domain is indicated by $\partial\Omega$.

- **Dirichlet Boundary Conditions**:
  Also sometimes called boundary conditions of the first type, are boundary conditions that specify a value on the boundary of the domain. Non-homogeneous Dirichlet boundary can generally be written as
  $$\phi(\mathbf{x}) = g(\mathbf{x}), \quad \text{for } \mathbf{x} \in \partial\Omega. \tag{2.11}$$
  In the homogeneous case, so $g(\mathbf{x}) = 0 \, \forall \mathbf{x} \in \partial\Omega$, the Dirichlet boundary conditions is sometimes referred to as a vanishing boundary conditions

- **Neumann Boundary Conditions**:
  or second-type boundary conditions, are boundary conditions that specify a value to the derivative of the boundary. Generally, this is boundary conditions is expressed as
  $$\frac{\partial\phi(\mathbf{x})}{\partial\mathbf{n}}\phi(\mathbf{x}) = h(\mathbf{x}), \quad \text{for } \mathbf{x} \in \partial\Omega. \tag{2.12}$$
  $\mathbf{n}$ denotes unit vector pointing outward with respect to the boundary Ω. The homogeneous, so $h(\mathbf{x}) = 0 \, \forall \mathbf{x} \in \partial\Omega$, Neumann boundary condition is sometimes called the reflective boundary conditions.

- **Robin Boundary Conditions**:
  Robin boundary conditions are a weighted combination of the first two boundary conditions. Its non-homogenous case is therefore expressed as
  $$\left(a\frac{\partial\phi(\mathbf{x})}{\partial\mathbf{n}} + b\phi(\mathbf{x})\right) = k(\mathbf{x}), \quad \text{for } \mathbf{x} \in \partial\Omega, \tag{2.13}$$
  with $a, b \in \mathbb{C}$ being the weight coefficients. Similarly as before, the homogeneous case is given when $k(\mathbf{x}) = 0 \, \forall \mathbf{x} \in \partial\Omega$.

- **Sommerfeld Radiation Condition**:
  Often, the Sommerfeld radiation condition is used for the Helmholtz problem to introduce some damping to the problem. The damping is beneficial since it allows the Helmholtz problem to be controlled more easily. This means that the problems we will see later on, will be reduced. It is for example easier to construct an iterative Helmholtz solver that is wave number independent for the Helmholtz problem when the Sommerfeld radiation condition is applied to the whole boundary, compared to when the homogeneous Dirichlet boundary condition is applied to the Helmholtz problem. The Sommerfeld radiation condition is a more specific version of the Robin boundary conditions. Sometimes the Sommerfeld radiation conditions is also referred to as the absorbing boundary condition and it is represented as
  $$\left(\frac{\partial\phi(\mathbf{x})}{\partial\mathbf{n}} - ik\phi(\mathbf{x})\right) = 0, \quad \text{for } \mathbf{x} \in \partial\Omega, \tag{2.14}$$
  where i is the imaginary number and k is the wave number in the Helmholtz equation.

2.3 Analytical Solution of the One Dimensional Helmholtz Equation

In this section we will give the analytical solution to the 1D non-homogeneous Helmholtz equation with homogeneous Dirichlet boundary conditions. For convenience we will use $u(x)$ instead of $\phi(x)$. The boundary value problem is given by

$$\begin{cases} -\frac{d^2u(x)}{dx^2} - k^2u(x) = \delta(x - \frac{L}{2}), & \text{in } \Omega = [0, L] \subset \mathbb{R}. \\ u(x) = 0 & \text{for } x \in \partial\Omega. \\ k \in \mathbb{N}\backslash\{0\}. \end{cases} \tag{2.15}$$

Note that the harmonic source term is placed in the middle of the domain and note that k is independent of x. For some problems, a wave number which is dependent on x is chosen ($k = k(x)$). This is for example the case when modeling the Helmholtz equation in a non-homogeneous medium.

The boundary value problem of (2.15) can be expressed in terms of the Green's function $G(x, x')$. First we note that the Sturm-Liouville operator given by

$$\mathcal{L} = \frac{d}{dx}\left[p(x)\frac{d}{dx}\right] + q(x). \tag{2.16}$$

Setting $p(x) = -1$ and $q(x) = -k^2$, we find the Sturm-Liouville operator for the boundary value problem of (2.15).
Therefore, we can now write the boundary value problem with the Green's function, with a general location x' instead of $\frac{L}{2}$ for the source location, as

$$\begin{cases} \mathcal{L}(G(x, x')) = \delta(x - x'), & \text{in } \Omega = [0, L] \subset \mathbb{R}. \\ G(x, x') = 0 & \text{for } x \in \partial\Omega. \end{cases} \tag{2.17}$$

An eigenfunction ϕ must satisfy

$$\mathcal{L}(\phi) = -\lambda\sigma(x)\phi. \tag{2.18}$$

"Eigenfunctions corresponding to different eigenvalues are orthogonal with the weight $\sigma(x)$."
(i.e. ϕ_n and ϕ_m are orthogonal with weight $\sigma(x)$ for $\lambda_n \neq \lambda_m$).

We take $\sigma(x) = -1$. Thus (2.18) gives us the ODE

$$\frac{d^2\phi}{dx^2} + k^2\phi = -\lambda\phi, \tag{2.19}$$

$$\frac{d^2\phi}{dx^2} + \underbrace{(k^2 + \lambda)}_{\alpha}\phi = 0 \tag{2.20}$$

$$\frac{d^2\phi}{dx^2} = -\alpha\phi \tag{2.21}$$

Independent solution can often be obtained in the form of exponential, $\phi = e^{rx}$. When this expression is substituted into the ODE of (2.21) we find the polynomial

$$r^2 = -\alpha. \tag{2.22}$$

We can distinguish three different cases:

- $\alpha > 0$,
- $\alpha = 0$,
- $\alpha < 0$.

α can not be complex, since α needs to be real for the boundary value problem to have nontrivial solutions. The three cases are analysed separately.

First, $\alpha = 0$. This implies that $r = 0$, which means that we get the only trivial solutions due to the Dirichlet boundary conditions.

Second, $\alpha < 0$. This means that $r = \pm\sqrt{-\alpha}$, which suggests a solution of the form

$$\phi(x) = c_1 \cosh(\sqrt{-\alpha}x) + c_2 \sinh(\sqrt{-\alpha}x). \tag{2.23}$$

Applying the boundary conditions, $\phi(0) = \phi(L) = 0$, gives us that $\phi(x) = 0$. This is also a trivial solution.

Finally, we look at the case $\alpha > 0$. This means that $r = \pm i\sqrt{\alpha}$. This suggests that the general solution is given by

$$\phi(x) = c_1 \cos(\sqrt{\alpha}x) + c_2 \sin(\sqrt{\alpha}x). \tag{2.24}$$

Applying the boundary conditions we find that $c_1 = 0$ and $\sin(\sqrt{\alpha}x) = 0$. Thus we find the eigenfunctions and eigenvalues

$$\phi_n = c_2 \sin(\frac{n\pi x}{L}), \tag{2.25}$$

$$\alpha_n = \frac{n^2\pi^2}{L^2} \text{ , for } n = 1, 2, 3, \ldots . \tag{2.26}$$

But since $\alpha = k^2 + \lambda$, we find

$$\phi_n = \sin(\frac{n\pi x}{L}), \tag{2.27}$$

$$\lambda_n = \frac{n^2\pi^2}{L^2} - k^2 \text{ , for } n = 1, 2, 3, \ldots . \tag{2.28}$$

c_2 is an arbitrary multiplicative constant, so we can say $c_2 = 1$.

We will seek to solve (2.15) by using the method of eigenfunction expansion. This gives us that the solution $u(x)$ can be expressed as a series of sines in the following way

$$u(x) = \sum_{n=1}^{\infty} \alpha_n(x') \sin(\frac{n\pi}{L}x). \tag{2.29}$$

We also know from (2.17) that

$$u(x) = G(x, x'). \tag{2.30}$$

$G(x, x')$ can now be expressed, since we know that $\lambda_n \neq 0 \, \forall n \in \mathbb{N}\backslash\{0\}$. This give the following

$$u(x) = G(x, x') = \sum_{n=1}^{\infty} \frac{\phi_n(x)\phi_n(x')}{-\lambda_n \int_0^L \phi_n^2(x)\sigma \, dx}. \tag{2.31}$$

$$= \frac{2}{L} \sum_{n=1}^{\infty} \frac{\sin(\frac{n\pi}{2}) \sin(\frac{n\pi}{L}x)}{\left(\frac{n^2\pi^2}{L^2} - k^2\right)}, \text{ for } k^2 \neq \frac{n^2\pi^2}{L^2} \text{ and } n = 1, 2, 3, \ldots . \tag{2.32}$$

Now we have expressed the solution $u(x)$ to the boundary value problem of (2.15).

2.4 Dimensionless Helmholtz Model

Equation (2.15) can be made dimensionless. The goal is to map the problem onto the unit domain $[0, 1]$. Equation (2.15) uses the arbitrary domain $[0, L]$. To get a dimensionless model we introduce the new variable $\widehat{x}$ such that

$$\widehat{x} = \frac{x}{L}, \tag{2.33}$$

from which it follows that

$$\frac{d\widehat{x}}{dx} = \frac{1}{L}. \tag{2.34}$$

Now equation (2.15) can be written as

$$\begin{cases} -\frac{d^2u(\widehat{x})}{dx^2} - \widehat{k}^2 u(\widehat{x}) = L^2\delta(\widehat{x} - \frac{L}{2}), & \text{in } \Omega = [0,1] \subset \mathbb{R}. \\ u(\widehat{x}) = 0 & \text{for } \widehat{x} \in \partial\Omega. \\ \widehat{k} = Lk, \text{ with } k \in \mathbb{N}\backslash\{0\}. \end{cases} \tag{2.35}$$

If not otherwise specified, we will use the notation $\widehat{k} = k$ and $\widehat{x} = x$ for the following chapters.

2.5 Concluding Remarks and Summary

- Applying separation of variables on the homogeneous wave equation gives the Helmholtz equation.
- An analytical solution to the Helmholtz problem can be given using Green's functions and the method of eigenfunction expansion.
- The Sommerfeld radiation condition introduces minor damping to boundary value problem. Therefore, it is expected that it is easier to acquire a numerically scalable solver when the Sommerfeld radiation condition is applied to the boundary.
- A dimensionless Helmholtz model is constructed by introducing a new variable.

3 | Iterative Methods & Preconditioners

It becomes very difficult to solve the Helmholtz equation analytically in dimensions higher than one. Therefore, instead of solving the Helmholtz equation analytically, it is more convenient to solve the problem by using a numerical method. For the Helmholtz problem, the numerical methods that are used are the finite element method and the finite difference method. For the research in this report we will only focus on methods that use the finite difference method numerical scheme.
By nature of the equation, for large wave numbers that matrix of the numerical discretization becomes indefinite. But, as we will see in the next section, a large wave number also results in the size of the problem becoming large. For medium sized numerical problem, it is still possible to use direct numerical solution methods. In the case of the Helmholtz problem using direct numerical solution methods is not realistic anymore. Therefore, for the Helmholtz problem iterative solution methods are used. Direct solution numerical methods can still serve a purpose as they are often used as solvers for subdomain problems in domain decomposition methods and multigrid methods.

Before describing the structure of this chapter, it is important to give a few clear definitions of somewhat general terms.

- an *efficient* numerical solver is defined as a solver that has solutions very close to the exact solutions, i.e. the numerical solutions gives a good representation of the real problem.
- a *numerically scalable* numerical solver has two attributes. First, the number of iterations needed to reach convergence can not grow when the wave number increases. Second, the solver needs a time complexity with respect to the number of grid nodes of $\mathcal{O}(n)$.
- a *parallel scalable* numerical solver is defined as a solver where the number of iterations needed to reach convergence does not grow when the number of subdomains (or parallel processes) increases.

This chapter is constructed in the following way. In the first section an introduction ...

3.1 Introduction/Problems when solving Helmholtz numerically

When trying to solve the Helmholtz equation numerically there are two major problems that arise [38]. The first problem is that for large wave numbers a so-called pollution error appears in the numerical solution [12][15][16]. This pollution error has to do with a phase misalignment of the exact and numerical solution. To deal with this pollution error, the grid has to be kept very fine. Specifically, when a second order finite difference scheme is used the requirement $k^3h^2 \leq 1$, with k being the wave number and h the length of a mesh element in one direction in the grid, has to hold in order to avoid the pollution error [15]. Thus, as the wave number becomes large, the grid has to be refined further.
For second order accurate finite difference discretizations it is a rule of thumb to have a minimum of 10 grid points per wave length λ [31]. Therefore we define

$$\kappa = kh = \frac{2\pi}{10} \approx 0.625. \tag{3.1}$$

This rule of thumb hold for wave numbers that are not too large. For larger wave number it might be necessary to use 20 or 30 grid points per wave length. In general, this rule of thumb is not very good. In [31] it is found that even if κ is kept small this is not to avoid the pollution error for high wave numbers. Instead, the requirement $k^3h^2 \leq \epsilon$ is introduced. We will mostly use the requirement that $k^3h^2 \leq 1$ or the weaker requirement $\kappa < 1$. The rule of thumb should result in the pollution error being avoided enough and the problem size not becoming too large when k becomes large.
When solving the Helmholtz equation numerically there seems to be no way to avoid the pollution error besides refining the grid as the wave number increases. A numerical solver that can deal with this accuracy problem is called an efficient numerical solver.
The second problem that arises when trying to solve the Helmholtz equation numerically is a consequence of the first problem. Due to the grid being required to be refined when k increases the size of

the problem also increases and quickly becomes large. Since the size of the problem is large, iterative methods are used as numerical solvers for the Helmholtz problem. As the grid becomes finer the computation cost of most iterative solvers increases. Therefore, a numerically scalable iterative solver has to be used. In order for a solver to be called numerically scalable, the solver needs to have two properties. The first property the solver needs is that the number of iterations to reach convergence does not grow with the wave number. The second property is that the solver has a time complexity with respect to the number of grid nodes of $\mathcal{O}(n)$.
An iterative Helmholtz solvers that is efficient and almost scalable has been developed [39]. It is not fully scalable since the solver has a nonlinear time complexity with respect to the number of grid nodes. In the following subsection the Helmholtz models that are used in this thesis are given. In sections after that details about iterative methods and preconditioning is given in order to understand Helmholtz numerical solvers and how to make them efficient.

3.1.1 Helmholtz Models

The boundary value problem for the Helmholtz equation with constant wave number and the Sommerfeld radiation condition is given by:

$$\begin{cases} -\frac{d^2u(\mathbf{x})}{d\mathbf{x}^2} - k^2u(\mathbf{x}) = f(\mathbf{x}), & \text{in } \Omega \in \mathbb{R}^d. \\ \left(\frac{\partial u(\mathbf{x})}{\partial \mathbf{n}} - iku(\mathbf{x})\right) = 0, & \text{for } \mathbf{x} \in \partial\Omega, \\ k \in \mathbb{N}\backslash\{0\}, \end{cases} \tag{3.2}$$

where d is either 2 or 3, depending if the problem is a 2D or 3D problem. Unless stated otherwise, $d = 2$ and the Helmholtz equation has a constant wave number. For the boundary conditions we have either a Dirichlet boundary condition or a Sommerfeld radiation condition. The difficulty in using Sommerfeld radiation conditions is that the eigenvalues of the resulting system matrix can not be expressed in closed-form and spectral analysis becomes more difficult but it can provide useful heuristics [38]. For this reason, some of the analysis is performed on the 1D boundary value problem without the Sommerfeld radiation condition. The 1D boundary value problem with Dirichlet boundary conditions is

$$\begin{cases} -\frac{d^2u(x)}{dx^2} - k^2u(x) = f(x), & \text{in } \Omega \in \mathbb{R}. \\ u(x) = 0 & \text{for } x \in \partial\Omega, \\ k \in \mathbb{N}\backslash\{0\}. \end{cases} \tag{3.3}$$

And the 2D case is given by

$$\begin{cases} -\frac{d^2u(\mathbf{x})}{dx^2} - k^2u(\mathbf{x}) = f(\mathbf{x}), & \text{in } \Omega \in \mathbb{R}^2. \\ u(\mathbf{x}) = 0 & \text{for } \mathbf{x} \in \partial\Omega, \\ k \in \mathbb{N}\backslash\{0\}. \end{cases} \tag{3.4}$$

The boundary value problems with Dirichlet boundary conditions are the worst case scenarios. It is therefore useful to test solvers for this case.

3.2 Finite Difference Discretization

We will discuss the finite difference method in the 2D case for equation (3.2) and (3.4). First the domain has to be discretized. Suppose we discretize the domain $\Omega = [0,1] \times [0,1]$ uniformly with mesh size $h = 1/n$ with n being the number of mesh elements. We get the grid Ω_h, which has $(n+1)^2$ nodes including the boundary nodes and the discrete grid is defined as

$$\Omega_h = \{(x_i, y_j) |\, x_i = (i-1)h, y_j = (j-1)h;\, h = 1/n, 1 \leq i, j \leq n+1, n \in \mathbb{N}\}. \tag{3.5}$$

And since we also need a discretization of the physics we also have

$$u(x_i, y_j) \approx u^h(x_i, y_j) \text{ for } (x_i, y_j) \in \Omega_h,$$
$$f(x_i, y_j) \approx f^h(x_i, y_j) \text{ for } (x_i, y_j) \in \Omega_h,$$

A global ordering of the grid nodes is defined in order to construct a linear system formulation. The boundary nodes are included in linear system formulation. For the internal and boundary nodes x-lexicographical ordering is introduced. The nodes with coordinates (i, j) are assigned a global index by

$$I = i + (j-1)(n+1), \text{ for } 1 \leq i, j \leq n+1 \tag{3.6}$$

We let a central second order accurate finite difference scheme and the boundary conditions be applied to (3.2) to discretize it. We find the linear system with the discrete solution $\mathbf{u}^h$

$$A^h \mathbf{u}^h = \mathbf{f}^h. \tag{3.7}$$

The exact solution, evaluated on the grid nodes, is denoted by $\mathbf{u}^h_{ex}$.

On the grid Ω_h, we apply a second order accurate finite difference scheme to discretize the partial differential equation of equation (3.2) or (3.4). The finite difference approximation for the internal nodes of the PDE is given by

$$\frac{-u^h_{i,j-1} - u^h_{i-1,j} + 4u^h_{i,j} - u^h_{i,j+1} - u^h_{i+1,j}}{h^2} - k^2 u^h_{i,j} = f^h_{i,j}\,, \text{ for } 2 \leq i, j \leq n. \tag{3.8}$$

The 5-point stencil for the internal nodes is given by:

$$h^{-2} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 - k^2h^2 & -1 \\ 0 & -1 & 0 \end{bmatrix}.$$

For the 2D boundary value problem with Sommerfeld radiation conditions stelcels for the boundary nodes can be given. For the boundary nodes we have two cases:

- 4 corner nodes,
- remaining boundary nodes.

Starting with the remaining boundary nodes. If we, for example, look at $u^h_{1,j}$ with $2 \leq j \leq n$, then we find the stencil

$$h^{-2} \begin{bmatrix} 0 & -1 & 0 \\ 0 & 4 - k^2h^2 - 2ikh & -2 \\ 0 & -1 & 0 \end{bmatrix}.$$

In order to obtain symmetry of the linear system matrix A^h later on, a rescaling is often performed. This gives the following stencil

$$h^{-2} \begin{bmatrix} 0 & -\frac{1}{2} & 0 \\ 0 & 2 - \frac{k^2h^2}{2} - ikh & -1 \\ 0 & -\frac{1}{2} & 0 \end{bmatrix},$$

where we replace $f^h_{1,j}$ by $\frac{1}{2} f^h_{1,j}$ for $2 \leq j \leq n$. For the other boundary nodes of the remaining boundary nodes case, the stencils can be found in the same way.
For the 4 corner nodes case we do something similar. For example, for the corner node of $u^h_{1,1}$, the following stencil is found

$$h^{-2} \begin{bmatrix} 0 & -2 & 0 \\ 0 & 4 - k^2h^2 - 4ikh & -2 \\ 0 & 0 & 0 \end{bmatrix},$$

Also for these boundary nodes rescaling is performed. This give the following stencil for $u^h_{1,1}$

$$h^{-2} \begin{bmatrix} 0 & -\frac{1}{2} & 0 \\ 0 & 1 - \frac{k^2h^2}{4} - ikh & -\frac{1}{2} \\ 0 & 0 & 0 \end{bmatrix},$$

where we replace $f^h_{1,1}$ by $\frac{1}{4} f^h_{1,1}$. The same thing is done for the other corner boundary nodes.

For the 2D boundary value problem with Dirichlet boundary conditions (3.4) different stencils have to be defined for the boundary nodes. The stencil of the boundary nodes is then simply given by

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

with the value on the boundary replacing the corresponding value of $f^h_{i,j}$. In the case of using homogeneous Dirichlet boundary conditions the boundary value is 0 for the whole boundary, so all the boundary nodes get $f^h_{i,j}$ replaced by 0.

Now we want to define the size of A^h. Since we *do not* eliminate the boundaries from the linear system, we have that:

$$A^h \in \mathbb{C}^{(n+1)^2 \times (n+1)^2}, \text{ and } \mathbf{u}^h, \mathbf{f}^h \in \mathbb{C}^{(n+1)^2}$$

Using the stencils, details about the matrix A^h can be given. It is straight forward to give the linear system formulation of the boundary value problem with Dirichlet boundary conditions of equation (3.4).

$$A^h = \frac{1}{h^2} \begin{pmatrix} h^2 I_{n+1} & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \widehat{T}^h & -\widehat{I}^h & 0 & \cdots & \cdots & 0 \\ 0 & -\widehat{I}^h & \widehat{T}^h & -\widehat{I}^h & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & -\widehat{I}^h & \widehat{T}^h & -\widehat{I}^h & 0 \\ 0 & \cdots & \cdots & 0 & -\widehat{I}^h & \widehat{T}^h & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & h^2 I_{n+1} \end{pmatrix} \in \mathbb{R}^{(n+1)^2 \times (n+1)^2} \tag{3.9}$$

with

$$\widehat{I}^h = \begin{pmatrix} 0 & 0 & 0 \\ 0 & I^h_{n-1} & 0 \\ 0 & 0 & 0 \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}, \widehat{T}^h = \begin{pmatrix} h^2 & 0 & 0 \\ 0 & T^h & 0 \\ 0 & 0 & h^2 \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}, \tag{3.10}$$

$$T^h = \begin{pmatrix} 4-k^2h^2 & -1 & 0 & \dots & \cdots & 0 \\ -1 & 4-k^2h^2 & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & -1 & 4-k^2h^2 & -1 \\ 0 & \cdots & \cdots & 0 & -1 & 4-k^2h^2 \end{pmatrix} \in \mathbb{R}^{(n-1)\times(n-1)} \tag{3.11}$$

and I^h_{n-1}, I^h_{n+1} are identity matrices on $\mathbb{R}^{(n-1)^2\times(n-1)^2}$ and $\mathbb{R}^{(n+1)^2\times(n+1)^2}$, respectively. For this boundary value problem it is also possible to give a closed-form expression for the eigenvalues of the matrix (3.9).

The linear system formulation for the boundary value problem of equation (3.2) is given by

$$A^h = \frac{1}{h^2} \begin{pmatrix} \widetilde{I}^h & \dot{I}^h & 0 & \cdots & \cdots & \cdots & 0 \\ \dot{I}^h & \widetilde{T}^h & \dot{I}^h & 0 & \cdots & \cdots & 0 \\ 0 & \dot{I}^h & \widetilde{T}^h & \dot{I}^h & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \dot{I}^h & \widetilde{T}^h & \dot{I}^h & 0 \\ 0 & \cdots & \cdots & 0 & \dot{I}^h & \widetilde{T}^h & \dot{I}^h \\ 0 & \cdots & \cdots & \cdots & 0 & \dot{I}^h & \widetilde{I}^h \end{pmatrix} \in \mathbb{C}^{(n+1)^2\times(n+1)^2} \tag{3.12}$$

with

$$\widetilde{I}^h = \begin{pmatrix} \beta & -\frac{1}{2} & 0 & \dots & \cdots & \cdots & 0 \\ -\frac{1}{2} & \alpha & -\frac{1}{2} & 0 & \cdots & \cdots & 0 \\ 0 & -\frac{1}{2} & \alpha & -\frac{1}{2} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & -\frac{1}{2} & \alpha & -\frac{1}{2} & 0 \\ 0 & \dots & \dots & 0 & -\frac{1}{2} & \alpha & -\frac{1}{2} \\ 0 & \cdots & \cdots & \cdots & 0 & -\frac{1}{2} & \beta \end{pmatrix} \in \mathbb{C}^{(n+1)\times(n+1)}, \tag{3.13}$$

where $\beta = 1 - \frac{k^2h^2}{4} - ikh$ and $\alpha = 2 - \frac{k^2h^2}{2} - ikh$. And

$$\widetilde{T}^h = \begin{pmatrix} \alpha & -1 & 0 & \dots & \cdots & \cdots & 0 \\ -1 & \gamma & -1 & 0 & \cdots & \cdots & 0 \\ 0 & -1 & \gamma & -1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & -1 & \gamma & -1 & 0 \\ 0 & \dots & \dots & 0 & -1 & \gamma & -1 \\ 0 & \cdots & \cdots & \cdots & 0 & -1 & \alpha \end{pmatrix} \in \mathbb{C}^{(n+1)\times(n+1)}, \tag{3.14}$$

where $\gamma = 4 - k^2h^2$ and α is the same as before.
And finally,

$$\dot{I}^h = \begin{pmatrix} -\frac{1}{2} & 0 & 0 & \dots & \cdots & \cdots & 0 \\ 0 & -1 & 0 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & -1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & 0 & -1 & 0 & 0 \\ 0 & \dots & \dots & 0 & 0 & -1 & 0 \\ 0 & \cdots & \cdots & \cdots & 0 & 0 & -\frac{1}{2} \end{pmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}, \tag{3.15}$$

As already noted earlier, for the boundary value problem with Sommerfeld radiation conditions enforced on the boundary it is not possible to give a closed-form expression of the eigenvalues of the coefficient matrix (3.12).

For writing convenience the discretized domain Ω_h is from now on just denoted with Ω. Also we now denote the vectors $\mathbf{u}^h$ and $\mathbf{f}^h$ by **u** and **f**, respectively. Finally, the matrix A^h is denoted A for convenience. So the linear system of equation (3.7) is now written as

$$A\mathbf{u} = \mathbf{f}, \text{ on } \Omega. \tag{3.16}$$

For the boundary value problem of equation (3.2), which has the Sommerfeld radiation condition on the whole boundary, we find that the coefficient matrix A is indefinite real symmetric. For the boundary value problem of equation (3.3), which has homogeneous Dirichlet boundary conditions on the whole boundary, the coefficient matrix A is complex symmetric, but non-Hermitian.

For the 1D Helmholtz problem with Dirichlet boundary conditions (Equation (3.3)) we have already found that the exact eigenvalues and eigenfunctions are given by

$$\phi_n = \sin(n\pi x), \tag{3.17}$$
$$\lambda_n = n^2\pi^2 - k^2 \text{ , for } n = 1, 2, 3, \dots, k^2 \neq n^2\pi^2 \tag{3.18}$$

The discrete eigenvalues and eigenfunctions for the same boundary value problem are

$$\begin{aligned} \widehat{\lambda}_l &= \frac{1}{h^2}\left[2 - 2\cos(l\pi h) - k^2h^2\right], \\ l &= 1, 2, 3, \dots, n+1 \end{aligned} \tag{3.19}$$

and

$$\widehat{\phi}_i = \sin(i\pi\mathbf{x}), 1 \leq i \leq n+1 \tag{3.20}$$

where $\mathbf{x} = [x_i], 1 \leq i \leq n+1$ represents the grid vector on the 1D version of Ω_h.

For the 2D Helmholtz problem with Dirichlet boundary conditions (Equation (3.4)) the exact eigenvalue and eigenfunction expressions are given by

$$\phi_{n,m} = \sin(n\pi x)\sin(m\pi y) \text{ , for } n, m = 1, 2, 3, \dots \tag{3.21}$$
$$\lambda_{n,m} = n^2\pi^2 + m^2\pi^2 - k^2 \text{ , for } n, m = 1, 2, 3, \dots, k^2 \neq n^2\pi^2 + m^2\pi^2. \tag{3.22}$$

The related discrete eigenvalue are given by

$$\widehat{\lambda}_{i,j} = \frac{1}{h^2}\left[4 - 2\cos(i\pi h) - 2\cos(j\pi h) - k^2h^2\right] \quad i,j = 1,2,3,\ldots,n+1 \tag{3.23}$$

and the discrete eigenfunctions by

$$\widehat{\phi}_{i,j} = \sin(i\pi \mathbf{x})\sin(j\pi \mathbf{y}), 1 \leq i,j \leq n+1 \tag{3.24}$$

with **x** and **y** being the grid vectors from the 2D discrete domain Ω_h.

3.3 Direct Method

Most of the information from this section is from the Scientific Computing lecture notes by C. Vuik and D.J.P. Lahaye [32]. One way of solving the system of linear equations of equation (3.16) is by using direct solution methods. These methods are often the method of choice when dealing with problems of moderate size. Often, direct solution methods are used as subdomain solvers when using domain decomposition methods or multigrid methods.

The direct method consists of two stages. First, the system matrix A from equation (3.16) is decomposed into an upper triangular matrix U and a lower triangular matrix L where the diagonal of L are all 1's. The decomposition is done in such a way that $A = LU$.
After the decomposition, the linear system of equation (3.16) can be easily solved using a forward linear solve followed by a backward linear solve. The linear system is solved with the following algorithm

$$LU\mathbf{u} = \mathbf{f} \implies L\mathbf{y} = \mathbf{f}, \text{ followed by } U\mathbf{u} = \mathbf{y}. \tag{3.25}$$

This gives the solution **u** of equation (3.16). The computational cost of the LU direct method is $\mathcal{O}(\frac{2}{3}n^3)$. Because of this, the speed of this method highly depends on the size of the problem. It is for this reason that iterative methods are used when dealing with large systems of linear equations. Currently, the state-of-the-art iterative methods are preconditioned Krylov subspace methods.

3.4 Krylov Subspace Methods

In this section we study Krylov subspace methods as iterative methods for solving systems of linear equations. As with the previous section, a lot of information of this section comes from [32]. Krylov subspace iterative method can also be used for eigenvalue problem. Some popular eigenvalue algorithms using the Krylov subspace method are Arnoldi and Lonczos. Krylov subspace methods are designed to avoid matrix-matrix operations, since these operations are computationally costly. Instead, Krylov subspace methods rely on multiplying vectors by the matrix. For using Krylov subspace iterative methods, the sequence of iterations of is denoted by

$$\{\mathbf{u}^k\}_{k\geq 0} \text{ where } \mathbf{u}^k \to \mathbf{u} \text{ for } k \to \infty \tag{3.26}$$

For the Richardson iterative method, the iterative scheme can be written in terms of residuals $\mathbf{r}^k := \mathbf{f} - A\mathbf{u}^k$ as

$$\mathbf{u}^{k+1} = \mathbf{u}^k + M^{-1}\mathbf{r}^k,$$

with the initial guess $\mathbf{u}^0$ given, and where $A = M - N$ with M assumed to exist and be non-singular. With no additional information, it is recommended that the initial guess is zero [26]. Writing out the first few steps of the recursion shows that

$$\mathbf{u}^k \in \mathbf{u}^0 + \text{span}\left(M^{-1}\mathbf{r}^0, M^{-1}A(M^{-1}\mathbf{r}^0), ..., (M^{-1}A)^{k-1}(M^{-1}\mathbf{r}^0)\right)$$

The Krylov-space of dimension k of matrix A with initial residual r^0 is then defined as

$$K^k(A;\mathbf{r}^0) := \text{span}\left(\mathbf{r}^0, A\mathbf{r}^0, ..., A^{k-1}\mathbf{r}^0\right)$$

The most well known Krylov subspace iterative methods are CG(Conjugate Gradient), GMRES(generalized minimum residual) and Bi-CGSTAB(biconjugate gradient stabilized). These methods all have their requirements when they can be used and when they are optimal to use.

- CG: the matrix A should be symmetric positive definite (SPD). This method is one of the best iterative methods, but is limited due to its requirements.
- GMRES type methods: These iterative method can be used for general matrices. These methods have long recurrences, but they do have some optimality properties
- Bi-CG type methods: Applicable for general matrices. The methods have short recurrences, but do not have an optimality property.

Because the matrix A will not be SPD, the CG method can not be used and will therefore also not be discussed any further.

3.4.1 Bi-CGSTAB

The Bi-CG Stabilized algorithms was introduced in [8]. The method is a Krylov subspace method that can be used for general matrices. The precondtioned algorithm of the Bi-CGSTAB method can be found below. More details about preconditioned linear systems can be found in the following section. The information about Bi-CGSTAB given in this subsection is from [32]. More details about the method can also be found in [32].

Algorithm 1: *Preconditioned Bi-CGSTAB Algorithm*

$\mathbf{u}^0$ is an initial guess; $\mathbf{r} = \mathbf{f} - A\mathbf{u}^0$;
$\bar{\mathbf{r}}^0$ is an arbitrary vetor, such that $(\bar{\mathbf{r}}^0, \mathbf{r}^0) \neq 0,$ e.g. for $\bar{\mathbf{r}}^0 = \mathbf{r}^0$;
$\rho_{-1} = \alpha_{-1} = \omega_{-1} = 1$; $\mathbf{v}^{-1} = \mathbf{p}^{-1} = 0$;
for $i = 0, 1, 2, ...$ **do**
  $\rho_{\mathrm{i}} = (\bar{\mathbf{r}}^0)^T \mathbf{r}^{\mathrm{i}}$; $\beta_{\mathrm{i}-1} = (\rho_{\mathrm{i}}/\rho_{\mathrm{i}-1})(\alpha_{\mathrm{i}-1}/\omega_{\mathrm{i}-1})$;
  $\mathbf{p}^{\mathrm{i}} = \mathbf{r}^{\mathrm{i}} + \beta_{\mathrm{i}-1}(\mathbf{p}^{\mathrm{i}-1} - \omega_{\mathrm{i}-1}\mathbf{v}^{\mathrm{i}-1})$;
  $\hat{\mathbf{p}} = \mathbf{M}^{-1}\mathbf{p}^{\mathrm{i}}$;
  $\mathbf{v}^{\mathrm{i}} = \mathrm{A}\hat{\mathbf{p}}$;
  $\alpha_{\mathrm{i}} = \rho_{\mathrm{i}} / \left(\bar{\mathbf{r}}^0\right)^{\mathrm{T}} \mathbf{v}^{\mathrm{i}}$;
  $\mathbf{s} = \mathbf{r}^{\mathrm{i}} - \alpha_{\mathrm{i}}\mathbf{v}^{\mathrm{i}}$
  if $\|\mathbf{s}\| < \mathcal{O}^n$ where $n > 0$ is small, **then**
    $\mathbf{u}^{\mathrm{i}+1} = \mathbf{u}^{\mathrm{i}} + \alpha_{\mathrm{i}}\hat{\mathbf{p}}$;
  end if
  $\mathbf{z} = \mathrm{M}^{-1}\mathbf{s}$;
  $\mathbf{t} = \mathrm{A}\mathbf{z}$;
  $\omega_{\mathrm{i}} = \mathbf{t}^{\top}\mathbf{s}/\mathbf{t}^{\top}\mathbf{t}$;
  $\mathbf{u}^{\mathrm{i}+1} = \mathbf{u}^{\mathrm{i}} + \alpha_{\mathrm{i}}\hat{\mathbf{p}} + \omega_{\mathrm{i}}\mathbf{z}$;
  if $\mathbf{u}^{\mathrm{i}+1} < \mathcal{O}^n$ where $n > 0$ is small, **then**
    $\mathbf{r}^{\mathrm{i}+1} = \mathbf{s} - \omega_{\mathrm{i}}\mathbf{t}$;
  end if
end for

3.4.2 GMRES

The General Minimal RESidual (GMRES) method is based on the MINRES method. The details about the GMRES method come from [32].
Instead of using the Lanczos method, the GMRES algorithm uses the Arnoldi method. The GMRES algorithm can be found below.

Algorithm 2: *GMRES*

Choose $\mathbf{u}^0$ and compute $\mathbf{r}^0 = \mathbf{f} - A\mathbf{u}^0$ and $\mathbf{v}^1 = \mathbf{r}^0 / \left\|\mathbf{r}^0\right\|_2$,
for $j = 1, \ldots, k$ **do**
  $\mathbf{v}^{\mathrm{j}+1} = A\mathbf{v}^{\mathrm{j}}$;
  for $i = 1, \ldots, j$ **do**
    $h_{\mathrm{ij}} := \left(\mathbf{v}^{\mathrm{j}+1}\right)^\top \mathbf{v}^{\mathrm{i}}, \mathbf{v}^{\mathrm{j}+1} := \mathbf{v}^{\mathrm{j}+1} - h_{\mathrm{ij}}\mathbf{v}^{\mathrm{i}}$,
  end for
  $h_{\mathrm{j}+1,\mathrm{j}} := \left\|\mathbf{v}^{\mathrm{j}+1}\right\|_2, \mathbf{v}^{\mathrm{j}+1} := \mathbf{v}^{\mathrm{j}+1}/h_{\mathrm{j}+1,\mathrm{j}}$
end for
The entries of upper $k+1 \times k$ Hessenberg matrix $\bar{H}_k$ are the scalars h_{ij}.

3.5 Preconditioning

Typically, iterative methods are combined with preconditions to increase the speed of convergence. The speed of convergence of iterative methods depends of the condition number κ of the system matrix A. If the eigenvalues are not well clustered, the condition number will be large which would cause slow convergence. Preconditioning is used to cluster the eigenvalues more favorably and therefore speed up the iterative method.
In order to understand the following to section, knowledge about basic iterative methods is necessary because these are the building blocks preconditioning in a Krylov subspace context.

The preconditioner is a matrix M which is similar to the matrix A. By using a preconditioner the linear system of equations (3.16) is transformed into the linear system

$$M^{-1}A\mathbf{u} = M^{-1}\mathbf{f}. \tag{3.27}$$

The preconditioned matrix is defined as, $\hat{A} := M^{-1}A$. The preconditioner matrix, M or M^{-1}, has the following requirements:

- the eigenvalues of $M^{-1}A$ need to be distributed around 1,
- computing $M^{-1}\mathbf{f}$ should be low cost.

When the matrix A is SPD the condition number is given by

$$\kappa(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}. \tag{3.28}$$

If the matrix is not SPD, then the condition number is more difficult to define. As noted earlier, when the Sommerfeld radiation condition is used the coefficient matrix becomes complex symmetric, but non-Hermitian, which introduces difficulties. The matrix A being non-normal results the condition number of the eigenvector matrix being larger than one. For GMRES it has been shown that if the coefficient matrix A is non-normal but diagonalizable the convergence of A behaves the same as if A was normal. This means that the convergence rate of non-normal A for GMRES does not explicitly depend on its eigenvalues, but the eigenvalues do influence the rate of convergence significantly.

3.5.1 Complex Shifted Laplacian Preconditioner

One of the preconditioners studied for the Helmholtz problem is the Complex Shifted Laplacian Preconditioner (CSLP) [20] [21]. The CSLP is defined as

$$M_{CSLP} := -\nabla_h^2 - (\beta_1 + i\beta_2)k^2 I_h = -\Delta_h - (\beta_1 + i\beta_2)k^2 I_h, \quad \beta_1, \beta_2 \in [0, 1], \tag{3.29}$$

where Δ_h is the discrete Laplacian operator and i is the imaginary unit. β_1 and β_2 represent the real and complex shift in the preconditioner, where the preconditioner is optimal for $(\beta_1, \beta_2) = (1, 0.5)$ [22]. Using the CSLP greatly benefits the convergence behavior of the iterative method. The CSLP clusters the eigenvalues on the complex plane on a unit circle with the left side of the circle being the origin, if

$\beta_1 = 1$ [26]. When the wave number is small only some eigenvalues lay around the origin, but when the wave number increase the number of eigenvalues that go towards 0 increases. Consequentially, this makes the iterative solver with the CSLP not numerically scalable since the smallest eigenvalue keeps becoming smaller, which causes the number of iterations needed for convergence to keep increasing.
The matrix M_{CSLP} can be approximately or exactly inverted. In [20], [21] and [38] one multigrid iteration is used to approximately invert the preconditioning matrix.
Specifically, one $V(1,1)$ multigrid iteration is often used to approximately invert the preconditioning matrix. $V(1,1)$ means that the multigrid method uses a V type cycle with 1 pre-smoothing step and 1 post-smoothing step.
It should be noted that when $\beta_1 = 1$, we find that the smaller β_2 the fewer eigenvalues are close to the origin. And the fewer eigenvalues close to the origin is favorable for the convergence of the preconditioned Krylov method. But, also note that the smaller β_2 the harder it is to do an approximate inversion of the preconditioner using multigrid, i.e. the higher β_2, the longer it takes for the multigrid method to converge and find the approximate inverse of the preconditioning matrix [35] [26].

3.5.2 Deflation

Deflation preconditioning is a method of removing the smallest eigenvalues. The removing is done by deflating the small eigenvalues all the way to 0. Eigenvalues that are 0 do not harm the convergence behavior and are ignored when computing the condition number of the preconditioned matrix $\hat{A}$.

Details about the deflation method are given in the case of having a 1D boundary value problem. Assume we have the linear system of equation such as from equation (3.16), but then in 1D. In order to describe the deflation method the projection P_D is defined as

$$P_D = I - P = I - AZ\left(Z^T AZ\right)^{-1} Z^T, \quad Z \in \mathbb{R}^{(n+1)\times r}. \tag{3.30}$$

In the equation above I is the identity matrix, and the column space of the matrix Z is the deflation subspace. The deflation subspace is the space that is to be projected out of the residual. Furthermore, we assume that Z has rank r and $r \ll (n+1)$.
We also define the matrix $E \in \mathbb{R}^{r\times r}$ where $E = Z^T AZ$. The matrix P is a projection which projects an input vector $\nu \in \mathbb{R}^{n+1}$ to the deflation subspace. But we are not interested in the deflation subspace, we are interested in the null space of the deflation subspace. P_D projects on the null space of P. Thus P_D projects on the null space of the deflation subspace.
We know that another projector is defined as

$$P_D^T = I - ZE^{-1}Z^T A. \tag{3.31}$$

The solution u of equation (3.16) can be written as

$$\mathbf{u} = (I - P_D^T)\mathbf{u} + P_D^T\mathbf{u}. \tag{3.32}$$

Only $P_D^T\mathbf{u}$ needs to be computed, since $\left(I - P_D^T\right)\mathbf{u}$ is computed very easily as

$$\left(I - P_D^T\right)\mathbf{u} = ZE^{-1}Z^T A\mathbf{u} = ZE^{-1}Z^T\mathbf{f}.$$

We know that $P_D A = AP_D^T$, thus the deflated system is given by

$$P_D A\tilde{\mathbf{u}} = P_D\mathbf{f} \tag{3.33}$$

Using a Krylov subspace iterative method the solution $\tilde{\mathbf{u}}$ can now be found. Multiplying $\tilde{\mathbf{u}}$ by P_D^T and adding it to equation (3.32) gives the solution to the linear system of equation (3.16).
Note that the null space never enters the iterative method. Therefore, the zero eigenvalues have no influence on the convergence of the iterative method. This is the reason for using the deflation method.

Applying deflation preconditioning comes with some difficulties. Deflation preconditioning is often not used in combination with the GMRES method, since the deflation method requires the original system matrix to be positive semi definite and self-adjoint. For the Helmholtz problem these requirements do not hold. Because the requirements do not hold it could be that the projections are ill-defined. Therefore, conditions for the matrix A and AZ can be given such that deflation is possible when the system matrix A is not positive semi definite and self-adjoint [29]. The following theorem is found in [29]

Theorem 1 (Deflated GMRES)**.** *Let $A\boldsymbol{u} = \boldsymbol{f}$ be a linear system with A non-singular, $A \in \mathbb{C}^{(n+1)\times(n+1)}$, $b \in \mathbb{C}^{n+1}$ and Z a subspace of $\mathbb{C}^{(n+1)\times(n+1)}$ with dimension $\dim(Z) = r < n$. Furthermore, let $\theta_{Z,AZ} < \frac{\pi}{2}$, where $\theta_{Z,AZ}$ denotes the principle angle between the subspaces Z and AZ. Then the project matrix $P_D := P_D\boldsymbol{u} = (I - P)\boldsymbol{u} = \boldsymbol{u} - AZ\langle Z, AZ\rangle^{-1}\langle Z, \boldsymbol{u}\rangle$, $\boldsymbol{u} \in \mathbb{C}^{n+1}$ is well-defined. Moreover, for all initial guesses $\boldsymbol{u}_0$, the GMRES method applied to the deflated system is well-defined.*

Proof. The proof of this theorem can be found in [29], in section 3.3 theorem 3.9 □

To conclude, all that is needed in order to apply the deflated GMRES to the projection matrix P_D is that the original system matrix A is non singular and that the principle angle between the subspaces Z and AZ is less than $\frac{\pi}{2}$.
Because this theorem does not apply to Bi-CGSTAB Krylov method, and because there is no equivalent theorem for the Bi-CGSTAB, the Bi-CGSTAB iterative method will not be used or discussed any further. The application of deflation preconditioning, CSLP and domain decomposition is essential in the work of this study. Therefore, this study will continue by using the GMRES Krylov method as the basis of its iterative solvers.

With the general background information about the deflation given, it is now possible to choose the deflation vectors as the columns of $Z \in \mathbb{C}^{(n+1)\times r}$ for the Helmholtz problem. Many options are available for Z and the choice of the deflation vectors is crucial for the success of the deflation method.
The goal of deflation is to remove the eigenvalues corresponding to the r smallest eigenvalues from the solution subspace. It is therefore natural to pick the eigenvectors corresponding to these r smallest eigenvalues as columns of Z. Choosing the matrix Z as such, results in the r smallest eigenvalues becoming zero. Even though this choice of column of Z is effective, it is also expensive. Computing the eigenvectors corresponding to the r smallest eigenvalues is computationally hard to do for large matrices.
Instead of computing the eigenvectors exactly, it is also possible to compute approximations of the eigenvectors corresponding to the r smallest eigenvalues. One possible approximation of the eigenvectors are the constant deflation vectors [4]. The domain Ω is divided into subdomains and the eigenvectors are approximated with constant deflation vectors in each subdomain. Each column of the matrix Z will then have ones at the grid index of the corresponding subdomain and zero everywhere else.
Another possibility is to use linear deflation vectors. These require more computational work compared to the constant deflation vectors, but they approximate the eigenvectors of the subdomains better.
Besides eigenvalue-based deflation, it is also possible to construct physics-based deflation vectors or algebraic deflation vectors. Only eigenvalue-based deflation is discussed in this report.
In the following chapter domain decomposition as a preconditioner is discussed. Deflation and domain decomposition are methods with a lot of similarities.

3.5.3 Deflation-based preconditioner

In [38] several deflation preconditions for the Helmholtz problem are given and tested numerically. The goal in the article is to construct a preconditioner that is wave number independent. The preconditioners in the article mostly consist of a CSLP in combination with deflation using a coarse correction operator as the deflation matrix Z from equation (3.30). This means that $Z = I_{2h}^h$. Several different definition for I_{2h}^h varying in order and if a weight is included in the coarse correction operator are given in [38].
Combining Deflation and CSLP preconditioning results in having to solve the following linear system

$$M_{CSLP}^{-1} P_D A\mathbf{u} = M_{CSLP}^{-1} P_D \mathbf{f}. \tag{3.34}$$

When using a standard linear grid interpolation function the preconditioning scheme is referred to as a DEF-based preconditioner. In this case we have a interpolation function as

$$I_{2h}^h \left[u_{2h}\right]_i = \left\{ \begin{array}{ll} \left[u_{2h}\right]_{(i+1)/2} & \text{if } i \text{ is odd,} \\ \frac{1}{2}\left(\left[u_{2h}\right]_{(i)/2} + \left[u_{2h}\right]_{(i+2)/2}\right) & \text{if } i \text{ is even} \end{array} \right)$$

for $i = 1, ..., n$.

The most promising preconditioner from the article, based on the numerical results, is given when the coarse correction operator in 1D is defined as

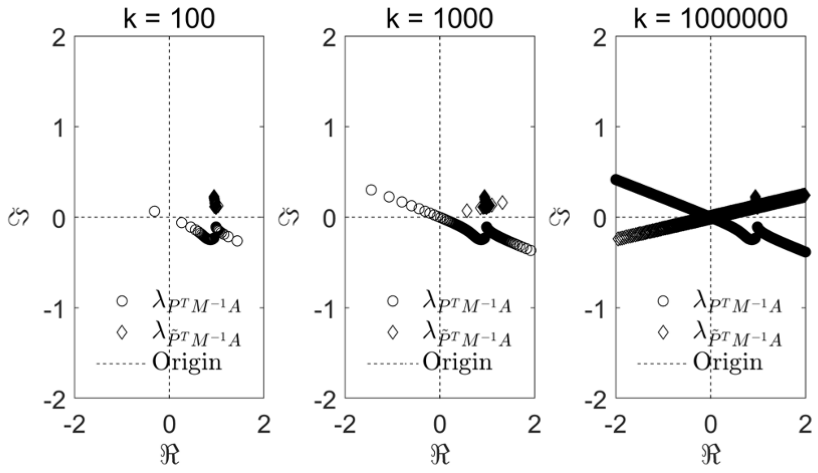
$$I_{2h}^{h}\left[u_{2h}\right]_i = \left\{ \begin{array}{cc} \left(\frac{1}{8}\left[u_{2h}\right]_{(i-2)/2} + \left(\frac{3}{4}-\varepsilon\right)\left[u_{2h}\right]_{(i)/2} + \frac{1}{8}\left[u_{2h}\right]_{(i+2)/2}\right) & \text{if } i \text{ is even,} \\ \frac{1}{2}\left(\left[u_{2h}\right]_{(i-1)/2} + \left[u_{2h}\right]_{(i+1)/2}\right) & \text{if } i \text{ is odd} \end{array} \right) \tag{3.35}$$

for $i = 1, ..., n$ and $\varepsilon > 0$. The definition for ε is given by

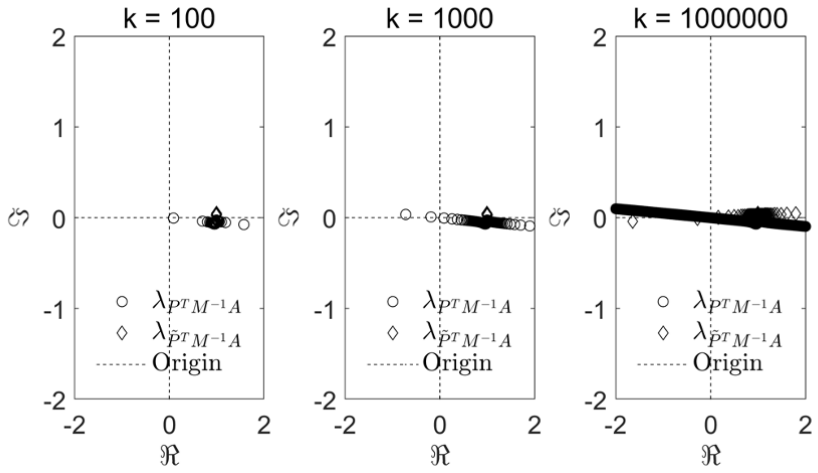
$$\varepsilon = 0.75 - \left(\cos(l\pi h) - \frac{1}{4}\cos(2l\pi h)\right). \tag{3.36}$$

The method of preconditioning which uses equation (3.35) is referred to as the the Adapted Preconditioned DEF scheme (APD). This method uses higher-order interpolation polynomials for the coarse correction operator, which is a quadratic approximation using the rational Bézier curve. Most often equation (3.36) is used to compute the optimal value for ε, but numerical tests are also performed using $\varepsilon = 0$.
After some numerical testing the article shows close to wave number independent convergence using the APD.

In order to keep track of the DEF-based scheme and the APD scheme we introduce that notation $\sim$ for the APD scheme. In Figure **??** the real and imaginary eigenvalues of the two deflation preconditioners (DEF and APD) are plotted for different wave numbers and different values of κ. One of the things we can conclude from the figures is that for very large wave numbers both preconditioner still have problems. This can seen since both preconditioned linear systems have eigenvalues that are not well clustered in the right column plots. Additionally, for the middle and left column we can see that the APD preconditioner performs better than the DEF precondtioner, since the eigenvalue clustering of the APD preconditioner is more favourable.



(a) $\kappa = 0.625$



(b) $\kappa = 0.3125$

Figure 3.1: The eigenvalues of the linear system with APD scheme indicated by $\tilde{P}M^{-1}A$ (diamond marker) and of the linear system with DEF-based scheme indicated by $PM^{-1}A$ (dot marker). In the top row $\kappa = 0.625$ and in the bottom row $\kappa = 0.3125$

Building on top of the APD scheme findings, a multilevel deflation preconditioner is constructed in [39]. The original APD is called a two-level deflation preconditioner, since it only uses a fine and a coarse grid. By introducing more layers of grids, the preconditioner becomes a multilevel deflation preconditioner.
It is important to note that instead of using a multigrid method on each level this article uses an exact solve on the coarsest level or when n is lower then a certain threshold and a few-GMRES iterations for the other levels. Earlier we said that due to the multigrid method β_2 had to be kept large, but when it is possible to use $\beta_2 = 1/k$, which provides optimal convergence when using GMRES as an iterative method and a direct method [34].
Using the multilevel deflation preconditioner the article finds by numerical experiments that the solver is almost numerically scalable and efficient. When Sommerfeld radiation conditions are used in the numerical experiments the article finds wave number independence for the number of iterations needed for convergence. It also find that the time complexity with respect to the number of grid nodes is somewhere between $\mathcal{O}(n^{1.5})$ and $\mathcal{O}(n)$.

3.6 Concluding Remarks and Summary

- To deal with the pollution error that occurs in the numerical solution, the grid has to be refined when the wave number increases.

- Numerical Helmholtz solvers often consist of a Krylov subspace iterative method combined with a preconditioner in order to improve number of iterations needed to reach convergence.
- GMRES is a suitable Krylov subspace iterative method and it allows the use of deflation preconditioning.
- CSLP + APD and CSLP + DEF preconditioning are are mainly the preconditioners that are discussed. Additionally, these show promising results.
- For very large wave numbers both preconditioners still show unfavourable clustering of the eigenvalues.

4 | Domain Decomposition Methods as a Preconditioner

Sometimes complications, such as too high computational costs, arise when a large problem on a big domain has to be solved. The basic idea of domain decomposition is that instead of solving the large problem, it may be necessary to solve many smaller problems on single subdomains a certain number of times. Due to the fact that domain decomposition methods solve many local subdomain problems, the method is well-suited for parallel computing.
A domain decomposition methods (DDM) can either be used as iterative method or as a preconditioner for iterative method. For this research we are interested in using DDMs as a preconditioner to accelerate the convergence of Krylov subspace methods. This means that a preconditioning matrix is constructed using domain decomposition methods and this preconditioning matrix approximates the coefficient matrix A^{-1} of the linear system of equations. In the books [10] [17] [19] [33] a lot has been written about the rapidly developing field of domain decomposition methods.

The first example of a domain decomposition method seems to be the alternating method by the mathematician H.A. Schwarz, which was published in 1870 [1]. Schwarz originally developed the method as a tool for proving the existence of a unique solution to the Laplace problem for an arbitrary domain [28]. In his method he used overlapping domains as in Figure 4.1. For this reason, overlapping domain decomposition methods are still referred to as Schwarz domain decomposition methods.
Domain decomposition methods are often divided into two classes. These classes are overlapping methods or also called Schwarz methods and non-overlapping or sometimes called substructuring methods.

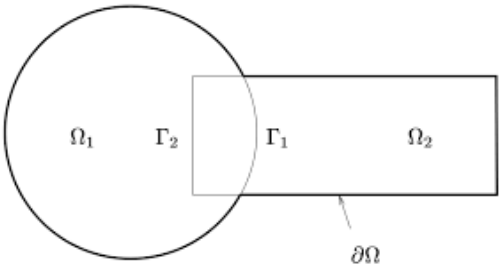


Figure 4.1: ...

This chapter is divided in the following way. In the first section Schwarz theory and Schwarz domain decomposition methods are discussed. After that, some brief information about non-overlapping domain decomposition methods, like Neumann-Neumann/FETI, are given. And finally, concluding remarks and a summary of this chapter is found in the final section.

4.1 Schwarz Domain Decomposition Preconditioner

With domain decomposition methods the full domain of a boundary value problem is decomposed into subdomains. As already mentioned in the introduction, Schwarz domain decomposition methods are characterized by having overlapping subdomains. It makes most sense to first discuss one-level Schwarz methods. After that we will discuss two-level DDM which are one-level DDMs with a coarse space added. Finally, different possible coarse spaces and their benefits are discussed. The order of the topics is like this since the topics naturally build on top of each other. It should be noted that the literature on domain decomposition methods are mainly written for finite element methods (FEM). We will mostly adhere to the FEM notation and only deviate when necessary.

Consider an elliptic partial differential equation problem with homogeneous Dirichlet boundary conditions on the domain Ω. Next a mesh $\mathcal{T}$ is introduced to the domain Ω and a finite element approximation is applied to the problem. The finite element approximation leads to a linear system formulation

$$A\mathbf{u} = \mathbf{f}, \tag{4.1}$$

with mesh size h and where **u** is the vector of which its coefficients are the discrete values of the approximate solution of the boundary value problem in the elements.

In the previous chapter iterative Krylov subspace solution methods were introduced to solve a finite difference method linear system of equations. There, it was shown that the convergence behavior of a Krylov subspace method strongly depends on the condition number of the system matrix A. In order to improve the distribution of the eigenvalues, and therefore to improve the condition number, a preconditioner can be used. Also in the previous chapter, the CSLP and deflation were discussed as a preconditioners to get a more favourable spectrum for the linear system.
In this chapter domain decomposition methods as a preconditioner are explored. This means that the different domain decomposition methods need to give definitions for the preconditioner operator M or M^{-1}. The benefit of using a domain decomposition method as a preconditioner is that that domain decomposition methods are often well-suited for parallel computing. Since we are interested in constructing a scalable efficient Helmholtz solver, parallel computing will be very beneficial.

For the sake of brevity and to avoid repetitiveness, the assumptions and starting point of the Schwarz domain decomposition method are given here. Consider the linear system as in equation (4.1), which arises from the finite element approximation of the elliptic partial differential equation, such as the Poisson problem, on the mesh $\mathcal{T}$ of domain Ω with homogeneous Dirichlet boundary conditions. Also consider a solution space V of basis functions on the mesh. The solution space is associate with the domain Ω.
Next, the polygonal domain $\Omega \subset \mathbb{R}^d$, with either $d = 1, 2, 3$, is partitioned, based on element splitting, into N non-overlapping uniform subdomains, which results in the set of subdomains $\{\Omega_i'\}_{i=1}^N$ with diameter H_i' and the maximum diameter of the subspaces is indicated by H'. The boundary of a non-overlapping subdomain where two non-overlapping subdomains meet is called the interface.
Now the subdomains are extended to a larger region, with $\{\Omega_i\}_{i=1}^N$ denoting the set of new subdomains, in such a way that the boundary of the extended subdomains do not cut through any elements of the mesh $\mathcal{T}$. The extension can be done be repeatedly adding a layer of elements. For the elements that are on the boundary we do the same thing, but we do not extend outside of Ω. The result of the extension is that we have constructed the set of N overlapping subdomains $\{\Omega_i\}_{i=1}^N$ with maximum diameter H. This now gives rise to N local overlapping meshes $\mathcal{T}_i$ on the overlapping subdomains Ω_i.
For the overlapping subdomains we introduce finite element solution spaces $\{V_i\}_{i=1}^N$ which consist of basis functions associated with the meshes $\{\mathcal{T}_i\}_{i=1}^N$. With K indicating the element in a mesh, we define

$$V_i = \left\{ u \in H_0^1(\Omega_i) \mid \quad u_{|_K} \in \mathbb{P}_1, K \in \mathcal{T}_i \right\}, \quad 1 \leq i \leq N. \tag{4.2}$$

Assume that the domain Ω is a 2D square domain, that the mesh has uniform square elements and that the subdomains are quasi-uniform. Figure 4.2 gives an illustration of a 2D partitioned grid with and without overlap. The degree of overlap is the amount of element layers of overlap there is and it is indicated by δ. Let I_i denote the indices of the nodes of the subdomains Ω_i. For the grid nodes we say that n indicates the number of nodes in the grid $\mathcal{T}$. For the subdomains we say that n_i indicates the number of nodes in the grid $\mathcal{T}_i$, for $i = 1, 2, ..., N$.
The overlapping domains Ω_i are all quasi-uniform squares, due to the extension, with square elements. Therefore, we can state the following In the case of no overlap ($\delta = 0$), n_i is the same for all subdomains and we only have the set of non-overlapping subdomains $\{\Omega_i'\}_{i=1}^N$. In the case that $\delta > 0$, n_i is not the same in all subdomains $\{\Omega_i\}$. n_i is not the same in all subdomains in this case because of the way that we extend the non-overlapping domain that are touching the boundary $\partial\Omega$. For either case of δ we know that $\sum_{i=1}^N n_i > n$ due to the shared nodes on the interface of the subdomains.
The system/stiffness matrix A from the linear system of equation (4.1) can also be transformed into a local system matrix for each subdomain. To do this, new operators have to be defined. Suppose that there exists extension operators

$$R_i^T : V_i \to V, \tag{4.3}$$

also sometimes referred to as prolongation operators. And the R_i matrices are often called restriction operators. The space V is assumed to decompose as

$$V = R_0^T V_0 + \sum_{i=1}^{N} R_i^T V_i. \tag{4.4}$$

Note that the decomposition will not necessarily be a direct sum of subspaces. This is because an element of V can often be found in more than one V_i space. V_i does not need to be a subspaces of V, but it is customary to refer to V_i as a subspace or local space. The space V_0, sometimes called the global space, is related to a coarse problem build on a coarse mesh and remaining spaces $\{V_i\}_{i=1}^N$ are related to the subdomains which followed from the partitioning we performed.
Now the local system matrices, sometimes referred to as the local operators, of subdomain Ω_i and local space V_i are given by

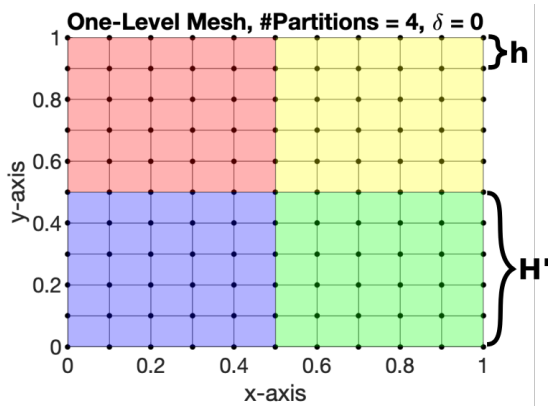
$$A_i = R_i A R_i^T, \quad \text{for } i = 1, ..., N \tag{4.5}$$

Here we can interpret the matrix A_i as the part of matrix A corresponding to the grid nodes in subdomain Ω_i.
Next we can introduce discrete projection-like operators as

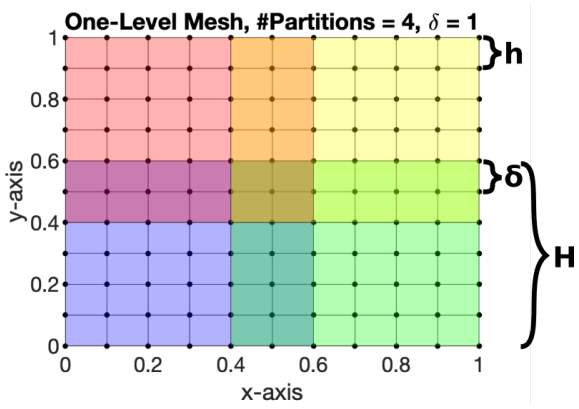
$$P_i = R_i^T A_i^{-1} R_i, \quad \text{for } i = 1, ..., N \tag{4.6}$$

One of the properties of these operators is that $\hat{P} = P_i A$ is a projection, i.e. $\hat{P}^2 = \hat{P}$.

Now that we have a set of subspaces and local problems, we can give a number of different Schwarz operators, which give us the different Schwarz domain decomposition methods. The Schwarz operators are defined by a polynomial of the operators $\{P_i\}$ without P_0.



(a) 2D grid domain image partitioned in 4 subdomains with no overlap. Note that the subdomains do share grid nodes on the interfaces of the partitions. $n_i = 6^2 = 36$ and $n = 11^2 = 121$



(b) 2D grid domain image partitioned in 4 subdomains with 1 grid layer of overlap $\delta = 1h$. $n_i = (6+1)^2 = 49$ and $n = 11^2 = 121$

Figure 4.2: Mesh image of non-overlapping and overlapping subdomains. The subdomains are indicated by the coloured areas. h indicates the mesh size, H' indicates the non-overlapping subdomain size, H indicates the maximum overlapping subdomain size and δ indicates the overlap. In (**b**) $\delta = 1$, which means that each non-overlapping subdomains is extended by 1 layer of elements. This leads to overlapping subdomains.

From [19] we also have the following assumptions for the local spaces. We say that δ_i measures the width of the regions $\Omega_i \backslash \Omega_i'$.

Assumption 1. *For $i = 1, ..., N$, there exists δ_i such that, if x belongs to Ω_i, then*

$$\textit{dist}(x, \partial\Omega \backslash \partial\Omega') \geq \delta_i,$$

for a suitable $j = j(x)$, possibly equal to i, with $x \in \Omega_j$. The maximum of the ratios H_i'/δ_i is denoted by

$$\frac{H}{\delta} = \max_{1 \leq i \leq N} \left\{ \frac{H_i'}{\delta_i} \right\}$$

Assumption 2 (Finite Covering)**.** *The partition $\{\Omega_i\}$ can be colored using at most N^c colors, in such a way that the subdomains with the same color are disjoint.*

4.1.1 One-Level Schwarz Domain Decomposition Methods

In the introduction the alternating Schwarz one-level domain decomposition method has already been mentioned. A lot of work on the abstract theory of the alternating Schwarz method is done by Lions in

[43]. The original domain decomposition methods were all methods that only used the local spaces created by the decomposition of the domain. Methods like that are called one-level domain decomposition methods. One-level Schwarz domain decomposition preconditioners were first introduced in [2] [3]. In this subsection we will discuss the one-level additive and multiplicative Schwarz domain decomposition methods, which can be used as preconditioners.

Multiplicative Schwarz Method

We will now give the definition of the multiplicative Schwarz (MS) method. The use of alternating Schwarz methods on boundary value problems has become to be known as the multiplicative Schwarz method. Considering the assumptions from above, we can now define the multiplicative Schwarz preconditioner as

$$M_{\mathrm{MS}}^{-1} = I - (I - P_N)(I - P_{N-1})...(I - P_1). \tag{4.7}$$

With $I \in \mathbb{R}^{(n+1)\times(n+1)}$ in identity matrix and $\{P_i\}_{i=1}^{N}$ are as defined in equation (4.6).
The MS method is a DD method that, due to its nature, has problems being combined with parallel computing [17]. For this reason the MS method is a method that will not be useful for the one-level Schwarz method. Without parallel computing, the MS method is fast than the additive Schwarz method [33]. When looking at two-level or multilevel Schwarz methods, the multiplicative method might again be useful.

Additive Schwarz Method

The remaining one-level Schwarz method is the one-level additive Schwarz (AS) method. It is also possible to give another variant of the one-level AS method called the restricted additive Schwarz (RAS) method. More details about the RAS method can be found in the subsubsection below this one.
The one-level AS method was first introduced in [5] and [7]. The preconditioner matrix for the one-level AS method is defined as

$$M_{\mathrm{AS}}^{-1} = \sum_{i=1}^{N} P_i = \sum_{i=1}^{N} R_i^T A_i^{-1} R_i. \tag{4.8}$$

This preconditioner uses exact solvers for the local spaces. One of the advantages of the AS method is that it is suitable for parallel computing. It is suitable for parallel computing because the local spaces can all be solved independently of each other. Because of the ability to combine the AS method with parallel computing, the AS method has gained popularity in the last two decades. Using the AS method as a preconditioner for a Krylov subspace method is sometimes referred to as the Krylov acceleration approach.

An upper bound for the condition number of the matrix $M_{\mathrm{AS}}^{-1}A$ is given in [17, p. 95]. Some necessary assumptions about the domain decomposition are given before the upper bound can be shown. For the condition number it is found that

$$\kappa(M_{\mathrm{AS}}^{-1}A) \leq C(\frac{1}{\delta^2 H^2}), \tag{4.9}$$

Where H is the maximum width of the subdomains Ω_i, for $i = 1, ..., N$, and δ is the overlap between the subdomains, where if δ is 0 there is no overlap and if δ is $H'/2$ the overlap is half the maximum width of the non-overlapping subdomains. The rate of convergence of the AS method combined with a Krylov method increase as that overlap of the subdomains increase up until some point. When the overlap becomes roughly equal to or larger than $H'/2$ rate of convergence decreases again. This happens because the number of colors N^c of Assumption 2 increases when the overlap becomes too large.

Restricted Additive Schwarz Method

The restriction/extension operator of eq. (4.5) can be seen as the sum of two terms $R_i = R_i^0 + R_i^c$, where R_i^0 only includes the non-overlapping nodes and R_i^c only includes the nodes of the overlapping area [17]. Using this, Cai and Sarkis introduced a new AS preconditioner [11] called the restricted additive Schwarz (RAS) method, which is defined as

$$M_{\mathrm{RAS}}^{-1} = \sum_{i=1}^{N} P_i = \sum_{i=1}^{N} (R_i^0)^T A_i^{-1} R_i \tag{4.10}$$

The benefit of this new preconditioner is that when a parallel implementation is used for this preconditioner the communication cost can be lowered. This is because computing $(R_i^0)u$ does not involve data exchange with the processors of the neighbouring subdomains.

4.1.2 Two-Level Additive Schwarz Method

It has been shown that the one-level Schwarz domain decomposition method are not numerically scalable with respect to the number of subdomains. Additionally, it is quite easy to see from (4.9) that if H decreases, due to the increase in the number of subdomains, that the upper bound for the condition number increases. By using a two-level Schwarz domain decomposition preconditioner, which includes a solve of a coarse space the size of the number of subdomains, the scalability problem can be avoided [33].
Two-level Schwarz methods are given in term of partitions of the domain Ω into subdomains Ω_i, and by a coarse shape-regular mesh $\mathcal{T}_H$. The only requirement for the elements of $\mathcal{T}_H$ is that the elements should have a diameter on the order of H_i if the element intersects subdomain Ω_i. Besides this requirement, the coarse mesh can be independent of the fine mesh $\mathcal{T}$. We introduce the finite element space V_0, which is defined is for now defined as

$$V_0 = \left\{ u \in H_0^1(\Omega) \mid \quad u_{|_K} \in \mathbb{P}_1, K \in \mathcal{T}_H \right\} \tag{4.11}$$

As already mentioned, by introducing a coarse space to the one-level AS preconditioner the convergence rate of the algorithm can be made independent of the number of subdomains[14]. The one-level AS method with a coarse space is what is called a two-level additive Schwarz method. It is also possible to have a two-level hybrid preconditioner, which combines an additive and multiplicative Schwarz method in a two-level Schwarz preconditioner [24]. With regards to parallel computing, the two-level hybrid method will be the most relevant method. Finally, it is also possible to construct a two-level MS preconditioners, which is quite similar to a multigrid method.

The construction of the coarse space can be done in several different ways. The most obvious way is to create a two level mesh by generating a coarse mesh, partitioning the coarse grid and refining the grid inside the partitions. This way a nested two level mesh is created. Figure 4.3 gives an illustration of a nested two-level mesh.

The one-level overlapping Schwarz methods were originally extended to the two-level form by [5] [6] [7] [41]. The Schwarz theory for the nested two-level grids, as in Figure 4.3, for the two-level overlapping Schwarz method is first presented in [9] and more two-level Schwarz theory is discussed in [19].

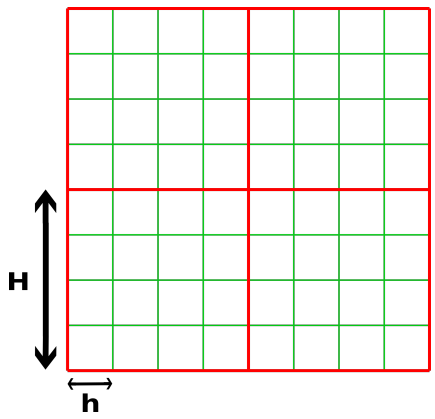


Figure 4.3: Illustration of a nested two-level grid. The coarse grid is indicated by red and it is a 3-by-3 grid. The fine grid is indicated with green and it is a 9-by-9 grid ($n = 8$).

The two-level additive Schwarz operator is of the form

$$M_{\mathrm{AS2}}^{-1} = \sum_{i=0}^{N} P_i = R_0^T A_0^{-1} R_0 + \sum_{i=1}^{N} R_i^T A_i^{-1} R_i, \tag{4.12}$$

where P_0 is the Schwarz operator of the coarse space V_0 and the remaining P_i are associated with the local problems on the subdomains from the set $\{\Omega_i\}_{i=1}^N$. A coarse interpolation operator is defined as

$$R_0^T : V_0 \to V, \tag{4.13}$$

which is obtained by interpolating the coarse functions onto the fine grid. Let $u \in V_0$, then we define

$$R_0^T u = I^h u, \tag{4.14}$$

with I^h being the interpolation function from the coarse mesh to the fine mesh. There are many different options for the interpolation functions. It is for example possible to use the interpolation function introduced in subsection 3.5.3.

The convergence behavior of the two-level additive Schwarz method can be analysed using the condition number of $M_{\mathrm{AS2}}^{-1}A$. In the case as in equation (4.12) and a coarse interpolation function as in equation (4.13), an upper bound for the condition number is given by

$$\kappa(M_{\mathrm{AS2}}^{-1}A) \leq C\left(1+\frac{H}{\delta}\right) \tag{4.15}$$

where H is the width of a subdomain and δ is the overlap between the subdomains. The upper bound of the condition number and the proof of equation (4.15) are found in [19, Theorem 3.13, p. 69].
The constant C depends on N_c, which is the maximum number of subdomains a node can be a part of and it is also the same N^c as in Assumption 2. For example, $N_c = 4$ in 4.2(**b**). C is independent of the mesh size in the coarse grid (indicated by H when we use a nested two-level grid, see Figure 4.3), the mesh size in the fine grid (indicated by h, see Figure 4.3), and the overlap of the subdomains (indicated by δ).
Equation (4.15) shows us that the two-level additive Schwarz method can be numerically scalable. This is because we can increase the problem size and at the same time keep the ratio of H/δ constant by increasing the amount of subdomains.

4.1.3 Two-Level Schwarz Methods with Different Coarse Spaces

Instead of using the coarse interpolation function as introduced in equation (4.13) it is also possible to use different coarse spaces. In the following subsection some other preconditioner with different coarse space definitions are given.

In [37] different preconditioners using domain decomposition preconditioning are constructed and tested for the Helmholtz equation. The preconditioners are two-level AS methods with two different coarse space definitions.
The first coarse space definition is just another grid coarse space with absorption. This is done in the following manner. First, a coarse mesh is constructed from which the operator and matrix R follows, which maps the nodes of the fine grid to to coarse grid. Instead of using $A_0 = RAR^T$ we now use $A_0 = RA_\varepsilon R^T$, where A_ε is the coefficient matrix for the boundary value problem with Sommerfeld radiation conditions, but with the PDE replaced by the following PDE

$$-\frac{d^2u(\mathbf{x})}{dx^2} - \left(k^2 + i\varepsilon\right)u(\mathbf{x}) = f(\mathbf{x}).$$

The other coarse space definition that is used is a Dirichlet-to-Neumann (DtN) coarse space, which is a coarse space that is based on local DtN eigenproblems on the subdomain interfaces. More details about this coarse space can also be found in [27] and [37].
The results of the work by M. Bonazzoli et al. show that the performance of the coarse spaces depends on the size of the coarse space size. For smaller coarse spaces the grid coarse space performs better and for larger coarse space the DtN coarse space performs better, where better means fewer iterations are needed compared to the other coarse space definition. The results are promising, but the iteration count does still (slowly) grow with the wavenumber.

GDSW Preconditioner

in [23] [24] the generalized Dryja-Smith-Widlund (GDSW) preconditioner is introduced, which is given by

$$M_{\mathrm{GDSW}}^{-1} = \sum_{i=0}^{N} P_i = \Phi^T A_0^{-1}\Phi + \sum_{i=1}^{N} R_i^T A_i^{-1} R_i, \tag{4.16}$$

where Φ is the energy-minimizing coarse space. Note that Φ is just the same as R_0 before.

The advantage of using GDSW type coarse spaces is that it gives us more freedom in constructing grids, since the GDSW coarse spaces can be used for unstructured grids. Another advantage of the

GDSW coarse spaces is that they are flexible in adding additional coarse functions. Furthermore, it would be sufficient to know the trace on the interface. The interior will be computed automatically.

Equation (4.16) is a two-level additive Schwarz preconditioner with exact solvers for the local spaces. The main ingredient of this precondition is the coarse space V_0. To construct the energy-minimizing coarse space, coarse basis function need to be constructed. For the GDSW preconditioner these coarse basis functions are defined by the introduction of a partition of unity on the interface Γ and an energy-minimizing extension to the interior.
The set Γ denotes the set of nodes on the interface of the decomposition. These nodes all have the property that they belong to more than one subdomain. Additionally, the set I denotes the set of the remaining nodes, which include the Dirichlet boundary nodes. The columns of Φ are the basis of the coarse space and we can write

$$\Phi = \begin{bmatrix} \Phi_I \\ \Phi_\Gamma \end{bmatrix} = \begin{bmatrix} -A_{II}^{-1} A_{\Gamma I}^T \Phi_\Gamma \\ \Phi_\Gamma \end{bmatrix}, \tag{4.17}$$

Where Φ_Γ is the restriction of the null space to the edges and vertices of the interface of the nonoverlapping partition. The set Γ can be divided further into K connected sets Γ_j. When nodes of edges and vertices are shared in the same set of subdomains they are combined into the set Γ_j. In the simple case of having structured rectangular subdomains the set of nodes with edges and vertices of a subdomain. For each set of interface nodes ,Γ_j, we construct a matrix Φ_{Γ_j} and the restriction matrix ,R_{Γ_j}, from Γ to Γ_j. With this the matrix

$$\Phi_\Gamma = \begin{bmatrix} R_{\Gamma_1}^T \Phi_{\Gamma_1} & \cdots & R_{\Gamma_M}^T \Phi_{\Gamma_M} \end{bmatrix} \tag{4.18}$$

is build and therefore Φ from Equation (4.17) is also given.
Now it is possible to compute the coarse coefficient matrix A_0 with

$$A_0 = \Phi^T A \Phi. \tag{4.19}$$

The condition number of the GDSW preconditioner is given by

$$\kappa\left(M_{\mathrm{GDSW}}^{-1} A\right) \leq C \left(1 + \frac{H}{\delta}\right) \left(1 + \log\left(\frac{H}{h}\right)\right)^2, \tag{4.20}$$

for general domains Ω partitioned into John domains [23][24]. This means that the condition number definitely holds for structured grids, since these always satisfy being John domains. John domains roughly means that the domain is such that it is possible to travel from one point in the domain to another point in the domain while not getting to close to the boundary.
With the following additional assumptions from [19] upper bound for the condition number can be improved. If the subdomains satisfy Assumption 4.3 from [19, p. 90] the term including a log function becomes first order. Additionally, the logarithmic term can be removed altogether using Theorem 3.13 from [19, p. 69].
Theorem 3.13 from [19, p. 69] gives the condition number upper bound of equation (4.15). This means that if the assumption necessary for Theorem 3.13 hold, then the GDSW preconditioner also has the condition number upper bound of equation (4.15), i.e.

$$\kappa(M_{\mathrm{GDSW}}^{-1} A) \leq C \left(1 + \frac{H}{\delta}\right) \tag{4.21}$$

It is shown that the dimensions of the energy-minimizing coarse spaces can be lowered [36]. Here, the article describes a method of constructing the coarse basis functions of the coarse space by using equivalence classes of nodes on the interfaces Γ. The article shows that the dimension of the coarse space can be reduced significantly without reducing the favorable upper bound for the condition number shown above.
In the case of using a 2D structured grid, this reduced dimension approach builds functions only corresponding to the vertices and not to the edges. So we can get rid of the edge functions.

4.2 Concluding Remarks and Summary

- Overlapping domain decomposition techniques for preconditioning of the Helmholtz problem provide the advantage of having a parallel scalable preconditioner.

- Overlapping domain decomposition preconditioners have been constructed and tested for the Helmholtz problem for different coarse space definitions. But no efficient and numerically scalable solver has yet been created.
- GDSW type coarse spaces have many advantages while also being parallel scalable.

5 | Parallel Computing

Parallel computing plays a major role in dealing with problems with large computational costs. The indefinite Helmholtz problem requires a large system of equations, especially in 2D and 3D. For large linear systems of equations the most appealing methods to solve those systems are iterative solution methods. Because of this, iterative solution methods are the main candidate to apply to parallel computing.
In this chapter we focus on standard algorithms in which as much parallelism as possible is implemented. This is different from designing an algorithm with parallelisation in mind. The literature in this chapter about parallel computing, high performance computing and MPI version 3 are found in [18] [25] [30].

A parallel computer is a computer that has several processors and can therefore perform several tasks simultaneously. These are tasks that are carried out simultaneously and not, as in multitasking, the rapid switching back and forth between tasks. Traditionally, a parallel computer is a supercomputer with thousands to hundreds of thousands of processors used for scientific computing, but today almost all computers are parallel; even the simplest PCs and laptops are equipped with multi-core processors.
The main reasons for using a parallel computer are:

- To have access to more memory storage,
- and to obtain a higher computational performance.

If a computer takes T_1 amount of time to run an algorithm, then in an ideal world parallelisation would allow p processing elements to run the same algorithm in $T_p = T_1/p$ amount of time. In practice, such efficiency can not be achieved due to factors such as overhead of processor communication, the work of the processors not being uniform or sequential sections in the algorithm. The definition speedup is defined as the factor of $S_p = T_1/T_p$.
In parallel computing often two types of scalability are used. These types of scalability are weak and strong scalability. We talk about strong scalability if a problem is partitioned over more and more processors and it shows almost perfect speedup, i.e. $S_p < p$ increases but S_p is close to p. Often in parallel computing literature sentences like “this problem scales up to 1000 processors." are used. Here, the authors are referring to strong scalability. It means that significant speedup can be achieved by partitioning the problem over at most 1000 processors, but little or no speedup is achieved after some threshold. Reasons for the speedup not increasing any further could be that the algorithm can not be partitioned any further or maybe the overhead communication cost becoming too large.
Weak scalability is used to describe the behavior of execution where the amount of data per processors stays the same, but the problem size and number of processors both grow.
For this research we are interested in weak scalability. This is because according to the domain decomposition theory we can construct a numerically scalable Helmholtz algorithm, i.e. a solver where we can increase the problem size by adding subdomains to the problem and keeping the size of the subdomains the same without the the upper bound of the condition number of the preconditioned system increasing.
Thus, successfully implementing the numerically scalable Helmholtz solver into a weakly scalable parallel program leads us to a Helmholtz solver that is parallel scalable and numerically scalable.

Parallel computing has several forms of parallelism. The main form of parallelism we are interested in is distributed computing. Distributed computing is a system of (identical) computers linked by some local area network, sometimes called the topology. The computers all have their own input/output subsystem, memory and CPU, but most importantly, with distributed memory each processor has its own address space. Having their own address space means that if two processors reference a variable y, then they get the variable from their own local memory. For the communication mechanisms for exchanging message between the computers the parallel virtual machine (PVM) and the message passing interface (MPI) are the best known libraries.

The main types of parallel architectures are the shared memory model, the data parallel models, also sometimes called the single instruction multiple data (SIMD) models, and the distributed memory message-passing models.

Since we are interested in solving many locals problems we want to use a parallel architecture that makes use of this property. Shared memory models can not take advantage of the locality of data in problems. This makes shared memory models unsuitable for this research.
Both data parallel models and distributed memory message-passing models are referred to as distributed memory models. The difference between data parallel models and message-passing models is that message-passing models have no global synchronization of the parallel tasks. This means that the local tasks on a processing element can start whenever that required data to start is available.
In the applications area the parallel architecture of choice are distributed memory message-passing models using MPI [18]. Because this type of architecture also suits our needs the distributed memory message-passing models with MPI are used.

5.1 Message Passing Interface (MPI)

As already mentioned, one way of facilitating the communication mechanics of a distributed memory computer is by using MPI. MPI version 1.0 was released in June 1994 with MPI-1 being version 1.3 of MPI. The latest version (version 3) of MPI was released in 2012 [30] and MPI-3 is MPI version 3.1. Besides that MPI is the most used communication mechanics, an additional advantage of MPI is that the topology is often hidden from the user and no topology coding is required [18].
MPI uses all kinds of routines which can roughly be devided in:

- Process management,
- Point-to-point communication,
- Collective calls.

Here process management means that MPI manages the parallel environment and can construct subsets of processing elements. Point-to-point communication means the routines which includes the set of calls between two processing elements, like send and receive calls. Finally, collective calls means the routines in which all the processing elements are involved. This is for example a gather call, where one processors is instructed to collect the data from all the processing elements participating.
MPI allows for a portable, efficient and functional use of distributed computing. More details about the using MPI are found in [30].

5.2 FROSch & Trilinos

Trilinos is an open-source object-oriented software suite for solving complex engineering and science problems using robust, scalable, parallel solver algorithms. Besides that, the goal of the Trilinos framework is also to provide a platform for the development of the robust, scalable parallel solver algorithms and libraries. The software of packages insides the Trilinos library are for the most part written in C++ Trilinos has the ability to use many of the major parallel architectures, including distributed computing with MPI.
The software library FROSch (Fast and Robust Overlapping Schwarz) [42] is a subpackage of the the software package ShyLU, which is found inside the Trilinos software suite. The ShyLU package provides two types of solvers. The first one are the distributed memory parallel domain decomposition solvers. The other type are node-level directs solvers for the subdomains.

6 | Research Proposal

Arguably, this chapter of the literature study is the most important chapter of them all. The goal of this chapter is to summarize and combine some of the knowledge from the previous chapters and propose clear and structured research questions for the thesis research. In the first section numerical experiments are performed on test problems. After that some general findings and possible conclusions are given in section two. The research questions for the master thesis research are given in section three.

6.1 Test Problems

In order to get some experimental knowledge of some of the topics discussed in this report it is useful to perform some numerical experiments on easy test problems. The two test problems include topics from chapters above. The goal of this section is not to show new research, but to play around with the information of the earlier chapters.
The topic in the first test problem is domain decomposition preconditioners from Chapter 4. Specifically, a simple 2D Poisson problem is solved using GMRES and a one-level additive Schwarz preconditioner. The second test problem is about numerically solving the Helmholtz problem using a Helmholtz solvers from the literature.

6.1.1 One-Level additive Schwarz Method: 2D Poisson Problem with non-homogeneous Dirichlet Boundary Conditions on a Unit Square

To see initial results about the one-level Schwarz domain decomposition method, the following problem is used. The problem is a 2D Poisson's equation problem on the domain $\Omega = [0,1] \times [0,1]$, which is described by the following equation

$$-\nabla^2 u(x,y) = f(x,y) \text{ on } \Omega. \tag{6.1}$$

The problem is supplied with non-homogeneous Dirichlet boundary conditions

$$u(x,y) = u_0(x,y) \text{ on } \partial\Omega. \tag{6.2}$$

Now assume that the exact solution to this problem is given by $u_{ex}(x,y) = \sin(xy)$. This gives us that the source term $f(x,y)$ is given by

$$f(x,y) = (x^2 + y^2)\sin(xy). \tag{6.3}$$

And the boundary data is given by

$$u_0(x,y) = \begin{cases} 0 & \text{if } x = 0 \\ \sin(y) & \text{if } x = 1 \\ 0 & \text{if } y = 0 \\ \sin(x) & \text{if } y = 1 \end{cases} \tag{6.4}$$

After discretization of the domain and applying a finite difference approximation to the PDE we end up with a linear system of equations. The GMRES iterative method with a one-level AS preconditioner is used to solve the linear system.
In the table 6.1 the number of iterations for convergence are given. The problem size is kept constant and H of the subdomains is decreased and the relative residual have a tolerance level set to 10^{-7}. The non-overlapping subdomains are all extended by 1 grid layer, i.e. $\delta = 1$.

H ($\delta = 1$)	n^2 = 6400	n^2 = 25600
0.5	17	22
0.25	24	33
0.125	30	43
0.0625	45	50+

Table 6.1: Number of GMRES iterations for the test problem using a one-level AS preconditioner.

6.1.2 APD preconditioner with GMRES: 2D Helmholtz problem with constant wave number

The following test problem is a 2D indefintite Helmholtz problem with constant wave number and homogeneous Dirichlet boundary conditions. The linear system is constructed both with and without elimination of the boundary nodes.
To solve the linear system the GMRES iterative method is used with the ADP preconditioner from [38]. Note that in the tables the ADP preconditioner is denoted by $DEF(0)+CSLP$. The ADP preconditioner consists of a combination of the CSLP and the deflation preconditioner, of which more information can be found in Section 3.5.3. What is key to note here is that the deflation precondition uses higher-order interpolation polynomials for the coarse correction operator and a weight ε. This higher-order coarse correction operator is a quadratic approximation using the rational Bézier curve. From the literature we find that $\varepsilon = 0.01906$ when $\kappa = 0.625$.
The reasons for performing the numerical experiments for this test problem are twofold. Firstly, this test problem gives an introduction to dealing with the indefinite Helmholtz problem and the problems that arise when trying to solve the indefinite Helmholtz problem numerically. The second reason is that the test problem allows us to use the quadratic approximation using the rational Bézier curve from the deflation preconditioner. These higher-order coarse correction operators could be used as a coarse space for a two-level Schwarz domain decomposition method.

For the grid resolution the requirement $\kappa = 0.625$ is used. Additionally we have $(\beta_1, \beta_2) = (1, 1/k)$ if exact inversion is performed for the CSLP. In the numerical experiment only exact inversion is used for CSLP. The results of the numerical experiments are found in table 6.2 and table 6.3. In the tables the x denotes that no computations was performed. The reason for this is that the computation would take a long time.

k	n^2	No Precon	CSLP	DEF(0.01906)	DEF(0) + CSLP
5	64	9	5	7	3
10	256	31	6	14	3
25	1600	101	8	17	3
50	6400	361	10	17	4
75	14400	x	13	16	5
100	25600	x	13	17	4
125	40000	x	13	17	5

Table 6.2: Number of iterations for the Helmholtz test problem, with elimination of the boundary conditions, using different or no preconditioners ($\kappa = 0.625$). CSLP uses $(\beta_1, \beta_2) = (1, 1/k)$ and deflation uses higher order interpolation polynomials for the coarse correction operator with $\varepsilon = 0.01916$ or $\varepsilon = 0$.

k	n^2	No Precon	CSLP	DEF(0.01906)	DEF(0) + CSLP
5	64	9	5	7	4
10	256	31	6	23	3
25	1600	101	8	29	3
50	6400	361	10	33	3
75	14400	x	13	34	4
100	25600	x	13	36	3
125	40000	x	13	37	4

Table 6.3: Number of iterations for the Helmholtz test problem, without elimination of the boundary conditions, using different or no preconditioners ($\kappa = 0.625$). CSLP uses $(\beta_1, \beta_2) = (1, 1/k)$ and deflation uses higher order interpolation polynomials for the coarse correction operator with $\varepsilon = 0.01916$ or $\varepsilon = 0$.

6.2 Preliminary Findings and Conclusions

- For the numerical experiment it can be seen that the one-level AS preconditioner is not scalable. As the theory already said, in order for the domain decomposition preconditioner to be scalable a second level has to be introduced.
- The APD preconditioner seems to be almost scalable from our numerical results. This is in agreement with the literature.

6.3 Research Questions

The main goal of this master thesis project is to construct an efficient and scalable Helmholtz solver, which exhibits robustness with respect to the wave number. In this research we are interested in using two-level overlapping Schwarz domain decomposition methods with higher-order coarse spaces.

The main questions are:

1. Does a Helmholtz solver that uses a two-level additive Schwarz preconditioner combined with **first-order** grid coarse space show **numerical scalability** and **efficiency**?
    - How do coarse spaces using linear coarse basis function perform?
    - Does using Restricted Additive Schwarz introduce further improvements?
2. Does a Helmholtz solver that uses a two-level additive Schwarz preconditioner and a **higher-order** coarse space from the **deflation setting** show **numerical scalability** and **efficiency**?
    - What causes the solver to be inscalable if this is the case?
    - How much better does the Helmholtz solver perform when the **higher-order coarse space from the deflation setting** is used as the **multiplicative** coupling of the coarse level?
3. When using higher-order coarse spaces shows promising performance, does using a **GDSW-type coarse space**, improve the Helmholtz solver further?
    - Can higher-order coarse spaces be used in GDSW-type coarse spaces?
    - Does this solver show improved performance?
4. If time permits, does the numerically scalable and efficient Helmholtz solver show **parallel scalability** when it is transformed into a **parallel algorithm**? To do this we could use **FROSch** of **Trilinos**.
    - Do memory or communication problems arise?
    - How much does a restricted additive Schwarz preconditioner improve the parallel algorithm further?

Bibliography

[1] Hermann Amandus Schwarz. *Gesammelte mathematische abhandlungen*. Vol. 260. American Mathematical Soc., 1972. ISBN: 0828402604.

[2] Aleksandr Mikhailovich Matsokin and Sergei Vladimirovich Nepomnyaschikh. “The Schwarz alternation method in a subspace”. In: *Izvestiya Vysshikh Uchebnykh Zavedenii. Matematika* 10 (1985), pp. 61–66. ISSN: 0021-3446.

[3] Sergey V Nepomnyaschikh. “Domain decomposition and Schwarz methods in a subspace for the approximate solution of elliptic boundary value problems”. Thesis. 1986.

[4] Roy A Nicolaides. “Deflation of conjugate gradients with applications to boundary value problems”. In: *SIAM Journal on Numerical Analysis* 24.2 (1987), pp. 355–365.

[5] Olof Widlund and Maksymilian Dryja. “An additive variant of the Schwarz alternating method for the case of many subregions”. In: (1987).

[6] Maksymilian Dryja. “An additive Schwarz algorithm for two-and three-dimensional finite element elliptic problems”. In: *Domain decomposition methods* (1989), pp. 168–172.

[7] Maksymilian Dryja and Olof B. Widlund. “Chapter 16 - Some Domain Decomposition Algorithms for Elliptic Problems”. In: *Iterative Methods for Large Linear Systems*. Ed. by David R. Kincaid and Linda J. Hayes. Academic Press, 1990, pp. 273–291.

[8] H. A. van der Vorst. “Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems”. In: *SIAM Journal on Scientific and Statistical Computing* 13.2 (1992), pp. 631–644.

[9] Maksymilian Dryja and Olof B Widlund. “Domain decomposition algorithms with small overlap”. In: *SIAM Journal on Scientific Computing* 15.3 (1994), pp. 604–620. ISSN: 1064-8275.

[10] William D. Gropp Barry F. Smith Petter E. Bjorstad. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge: Cambridge University Press, 1996. ISBN: 0521602866.

[11] Xiao-Chuan Cai and Marcus Sarkis. “A Restricted Additive Schwarz Preconditioner for General Sparse Linear Systems; CU-CS-843-97”. In: (1997).

[12] F. Ihlenburg and I. Babuska. “Finite element solution of the Helmholtz equation with high wave number .2. The h-p version of the FEM”. In: *SIAM JOURNAL ON NUMERICAL ANALYSIS* 34.1 (1997), pp. 315–358.

[13] David E Keyes. “Parallel numerical algorithms: An introduction”. In: *Parallel Numerical Algorithms*. Springer, 1997, pp. 1–15.

[14] Barry F Smith. “Domain decomposition methods for partial differential equations”. In: *Parallel Numerical Algorithms*. Springer, 1997, pp. 225–243.

[15] A. Deraemaeker, I. Babuska, and P. Bouillard. “Dispersion and pollution of the FEM solution for the Helmholtz equation in one, two and three dimensions”. In: *INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING* 46.4 (1999), pp. 471–499.

[16] K. Gerdes and F. Ihlenburg. “On the pollution effect in FE solutions of the 3D-Helmholtz equation”. In: *COMPUTER METHODS IN APPLIED MECHANICS AND ENGINEERING* 170.1-2 (1999), pp. 155–172.

[17] Alfio Maria Quarteroni and Alberto Valli. *Domain decomposition methods for partial differential equations*. Oxford University Press, 1999. ISBN: 0198501781.

[18] Yousef Saad. *Iterative methods for sparse linear systems*. SIAM, 2003. ISBN: 0898715342.

[19] Andrea Toselli and Olof Widlund. *Domain decomposition methods-algorithms and theory*. Vol. 34. Springer Science & Business Media, 2004. ISBN: 3540206965.

[20] Yogi A Erlangga, Cornelis W Oosterlee, and Cornelis Vuik. “A novel multigrid based preconditioner for heterogeneous Helmholtz problems”. In: *SIAM Journal on Scientific Computing* 27.4 (2006), pp. 1471–1492. ISSN: 1064-8275.

[21] Yogi A Erlangga, Cornelis Vuik, and Cornelis W Oosterlee. “Comparison of multigrid and incomplete LU shifted-Laplace preconditioners for the inhomogeneous Helmholtz equation”. In: *Applied numerical mathematics* 56.5 (2006), pp. 648–666. ISSN: 0168-9274.

[22] Martin B van Gijzen, Yogi A Erlangga, and Cornelis Vuik. “Spectral analysis of the discrete Helmholtz operator preconditioned with a shifted Laplacian”. In: *SIAM Journal on Scientific Computing* 29.5 (2007), pp. 1942–1958. ISSN: 1064-8275.

[23] Clark R Dohrmann, Axel Klawonn, and Olof B Widlund. “A family of energy minimizing coarse spaces for overlapping Schwarz preconditioners”. In: *Domain decomposition methods in science and engineering XVII*. Springer, 2008, pp. 247–254.

[24] Clark R Dohrmann, Axel Klawonn, and Olof B Widlund. “Domain decomposition for less regular subdomains: Overlapping Schwarz in two dimensions”. In: *SIAM journal on numerical analysis* 46.4 (2008), pp. 2153–2168. ISSN: 0036-1429.

[25] Victor Eijkhout. *Introduction to high performance scientific computing*. Lulu. com, 2010. ISBN: 1257992546.

[26] O. G. Ernst and M. J. Gander. “Why it is difficult to solve Helmholtz problems with classical iterative methods”. In: ed. by T. Y. Hou et al. Vol. 83. Springer Verlag, 2012, pp. 325–363.

[27] Lea Conen et al. “A coarse space for heterogeneous Helmholtz problems based on the Dirichlet-to-Neumann operator”. In: *Journal of Computational and Applied Mathematics* 271 (2014), pp. 83–99. ISSN: 0377-0427.

[28] Martin J Gander and Gerhard Wanner. “The origins of the alternating Schwarz method”. In: *Domain decomposition methods in science and engineering XXI*. Springer, 2014, pp. 487–495.

[29] André Gaul. “Recycling Krylov subspace methods for sequences of linear systems”. In: (2014).

[30] William Gropp, Ewing Lusk, and Anthony Skjellum. *Using MPI: portable parallel programming with the Message-Passing-Interface*. Third edition. Cambridge, MA: The MIT Press, 2014. ISBN: 0262326604 9780262326605.

[31] A.H. Sheikh. “Development Of The Helmholtz Solver Based On A Shifted Laplace Preconditioner And A Multigrid Deflation Technique”. Thesis. 2014.

[32] C. Vuik and D. J. P. Lahaye. *Scientific computing : WI4201*. Delft: TU Delft, 2014.

[33] Victorita Dolean, Pierre Jolivet, and Frédéric Nataf. *An introduction to domain decomposition methods: algorithms, theory, and parallel implementation*. SIAM, 2015. ISBN: 1611974054.

[34] Martin J Gander, Ivan G Graham, and Euan A Spence. “Applying GMRES to the Helmholtz equation with shifted Laplacian preconditioning: what is the largest shift for which wavenumber-independent convergence is guaranteed?” In: *Numerische Mathematik* 131.3 (2015), pp. 567–614. ISSN: 0945-3245.

[35] Pierre-Henri Cocquet and Martin J Gander. “How large a shift is needed in the shifted Helmholtz preconditioner for its effective inversion by multigrid?” In: *SIAM Journal on Scientific Computing* 39.2 (2017), A438–A478. ISSN: 1064-8275.

[36] Clark R Dohrmann and Olof B Widlund. “On the design of small coarse spaces for domain decomposition algorithms”. In: *SIAM Journal on Scientific Computing* 39.4 (2017), A1466–A1488. ISSN: 1064-8275.

[37] M. Bonazzoli et al. “Two-level Preconditioners for the Helmholtz Equation”. In: *Lecture Notes in Computational Science and Engineering*. Vol. 125. Springer Verlag, 2018, pp. 139–147.

[38] Vandana Dwarka and Cornelis Vuik. “Scalable convergence using two-level deflation preconditioning for the helmholtz equation”. In: *SIAM Journal on Scientific Computing* 42.2 (2020), A901–A928. ISSN: 1064-8275.

[39] Vandana Dwarka and Cornelis Vuik. “Scalable multi-level deflation preconditioning for the highly indefinite Helmholtz equation”. In: (2020). ISSN: 1389-6520.

[40] Vandana Dwarka et al. “Towards accuracy and scalability: Combining Isogeometric Analysis with deflation to obtain scalable convergence for the Helmholtz equation”. In: *Computer Methods in Applied Mechanics and Engineering* 377 (2021), p. 113694. ISSN: 0045-7825.

[41] Maksymilian Dryja and Olof B Widlund. “Towards a unified theory of domain decomposition algorithms for elliptic problems”. In: *Third international symposium on domain decomposition methods for partial differential equations*, pp. 3–21.

[42] A. Heinlein et al. “FROSch: A Fast And Robust Overlapping Schwarz Domain Decomposition Preconditioner Based on Xpetra in Trilinos”. In: *Lecture Notes in Computational Science and Engineering*. Ed. by R. Haynes et al. Vol. 138. Springer Science and Business Media Deutschland GmbH, pp. 176–184.

[43] Pierre-Louis Lions. “On the Schwarz alternating method. I”. In: *First international symposium on domain decomposition methods for partial differential equations*. Vol. 1. Paris, France, p. 42.