Master's Thesis

Construction of a Combined Preconditioner for the Helmholtz Problem

J.M. Tang

Shell International Exploration & Production, Rijswijk
Exploratory Research

Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Department of Applied Mathematics
Numerical Analysis Group

November 2003 – August 2004

Construction of a Combined Preconditioner for the Helmholtz Problem

J.M. Tang

Delft, August 2004

Thesis submitted to the Delft University of Technology
for the degree of Master of Science / Wiskundig Ingenieur.

The project has been carried out at
Delft University of Technology
and
Shell International Exploration and Production
under supervision of
dr. ir. C. Vuik and dr. W.A. Mulder.

Members of the Master Committee:

prof. dr. ir. P. Wesseling [TU Delft]
dr. ir. C. Vuik [TU Delft]
dr. W.A. Mulder [Shell Rijswijk]

Version: 1.0.
Compiled at: August 5, 2004.

To my parents Lisa & Cheung,
To my dear friend Siu Mei,
Whose support,
Encouragement,
And faithful prayers
Made this work possible

Acknowledgments

This work is my master's thesis of the study Applied Mathematics. The graduation work is done at the department of Numerical Analysis at Delft University of Technology in colloboration with Shell International Exploration and Production at Rijswijk.

The first part of this work consisted of a literature study of three months which has been finished with a report [48] and a presentation. During the literature study we developed some ideas for further research, which are worked out in the second part of the project. In these last six months, we have dealt with various aspects on, among other things, preconditioners and spectral analysis. The outcome is this master's thesis, which consists of several theoretical and numerical observations and results.

I want to thank a lot of people who where involved in my research and the writing of my thesis.

First of all, I would like to express my gratitude to dr. ir. Kees Vuik for his more than excellent supervising and encouragements during the graduation work. The weekly appointments were very pleasant and useful for the improvement of the research. It also motivated me to do this work accurately and with enthusiasm.

Special thanks I owe to dr. Wim Mulder from Shell Rijswijk who supervised me. He has given me some of his valuable time that contributed to the completion of the research at Shell. His help, advices and ideas were very useful for this final thesis.

Then, I would like to thank prof. dr. ir. Piet Wesseling and ir. Yogi Erlangga for their contributions to the research project. Also, I would like to thank my room mate ir. Sander van der Pijl for his help, for his lovely music and for the nice conversations during the last months of the project. Moreover, I thank my colleagues of Shell and of TU Delft, especially Sander van Veldhuizen, Saad Ibrahim and Jelle Hijmissen for the pleasant daily coffee, lunch breaks and discussions about international politics.

Moreover, I thank all members of the Master committee for their reading of this thesis, especially dr. ir. Kees Vuik and dr. Wim Mulder, who were also careful readers of my intermediate papers during the research project.

Last but not the least, a special word of thank towards my friends and family, who gave me all the imaginable love and support.

Jok Tang
Delft, July 2004

Summary

The Helmholtz equation combined with suitable boundary conditions results in the Helmholtz problem (HP). The efficient solving of this problem on very large grids is very important for companies like Shell. The Helmholtz problem is used for seismic investigations of the earth's crust. The results can be used to determine the position of various layers. Thereafter, possible locations of oil or gas reservoirs can be predicted.

Direct solvers are relatively expensive in practical 3-dimensional Helmholtz problems, whereas iterative methods with suitable preconditioners are attractive. However, until now there does not exist efficient preconditioners which can deal with this Helmholtz problem.

In this research project we consider two preconditioners, based on separation of variables (SoV) and complex shifted Laplace (CSL) methods, in more detail. We examine their iterative behaviour in test problems and also their spectral and convergence properties. To restrict the computational time in our test runs, we consider only small 2-dimensional problems.

We pay special attention to the preconditioner based on separation of variables. The whole method and the underlying linear algebra ideas are described. Moreover, some ideas improving this preconditioner are worked out and tested in more detail.

Investigating the properties of the preconditioners CSL and SoV is important to understand why the methods fail. In this thesis, this is carried out by comparing the eigenvalues and the corresponding eigenvectors of the original and the preconditioned systems. After that, several attempts are taken to combine these precondtioners into a new preconditioner, which are expected to get rid of the bad eigenvalues of the original system and in that way would lead to fast convergence for the Helmholtz problem. Unfortunately, considering the results of our test runs, the combined preconditioners do not improve the convergence rate, comparing to the SoV and CSL preconditioners.

Samenvatting

De Helmholtz vergelijking gecombineerd met geschikte randvoorwaarden leidt tot het Helmholtz probleem (HP). Het efficiënt oplossen van dit probleem op zeer omvangrijke roosters is van cruciaal belang voor bedrijven als Shell. Het Helmholtz probleem speelt een rol bij het seismisch onderzoek van de aardkorst. Het wordt gebruikt om de verschillende aardlagen in beeld te brengen, waarna geschikte locaties voor het vinden van olie en gas bepaald kunnen worden.

Vanwege de omvang van het Helmholtz probleem zijn exacte oplosmethoden relatief duur, terwijl iteratieve methodes met geschikte preconditioneringen in dit soort situaties aantrekkelijk zijn. Tot op heden is er echter nog geen goede preconditioner gevonden die de praktische Helmholtz problemen efficiënt kan oplossen.

In dit onderzoek zullen we twee preconditioners, gebaseerd op 'separatie der variabelen' (SoV) en op 'complex verschoven Laplace' (CSL) methoden, nader onder de loep nemen. We onderzoeken hun iteratief gedrag in testproblemen en we bekijken hun spectraal en convergentie eigenschappen. Om rekentijd te besparen beperken we ons tot relatief kleine 2-dimensionale problemen.

Speciale aandacht verdient de SoV preconditioner. De gehele methode en achterliggende linear algebra gedachten worden in detail beschreven en enkele ideeën om deze te verbeteren worden nader uitgewerkt en getest.

Het onderzoeken van de eigenschappen van de hierboven genoemde preconditioneringen is van groot belang om te kunnen begrijpen waarom de methoden niet voldoende efficiënt in gebruik zijn. In deze scriptie wordt dat uitgevoerd door de eigenwaarden en de corresponderende eigenvectors van de originele en de gepreconditioneerde systemen te vergelijken. Vervolgens worden diverse pogingen ondernomen om de twee preconditioneringen te combineren tot een nieuwe preconditioner, die in staat geacht wordt om grip te krijgen op de slechte eigenwaarden van het originele systeem en waarvan we derhalve verwachten dat het een goed convergentiegedrag zal vertonen ten aanzien van het Helmholtz probleem. Echter, het blijkt uit onze testruns dat de gecombineerde preconditioners niet efficienter presteren dan de oorspronkelijke CSL en SoV preconditioners.

Contents

List of Figures

List of Tables

List of Symbols

Latin Symbols

Symbol	*Meaning*
$\mathbf{A}$	coefficient matrix of the discrete Helmholtz problem
$\overline{\mathbf{A}}$	conjugate of matrix $\mathbf{A}$
$\hat{\mathbf{A}}$	modified $\mathbf{A}$ where $\tilde{k} = 0$ is taken
$\mathbf{B}$	modified $\hat{\mathbf{A}}$
$\mathbf{B}$	matrix obtained from the (positive) Laplace operator
c	background velocity in the layer
$\mathbf{D}$	diagonal of matrix $\mathbf{A}$
$\mathbf{D}$	block diagonal matrix in SoV technique
$\mathbf{D}_m$	subblock of block diagonal matrix $\mathbf{D}$
d	count coefficient
$\mathbf{f}$	source term vector
f	source term
f	frequency
G	minimal number of gridpoints per wavelength
$\mathbf{g}$	modified $\mathbf{f}$ as in the system $\mathbf{Dv} = \mathbf{g}$
$\mathbf{I}$	identity matrix
$\mathbf{K}$	diagonal matrix of the wavenumber of the domain
$\mathbf{K}_x$	diagonal matrix of k_x
$\mathbf{K}_y$	diagonal matrix of k_y
$\widetilde{\mathbf{K}}$	diagonal matrix of $\tilde{k}$
$\widetilde{\mathbf{K}}_{mod(i)}$	a modified diagonal matrix of $\tilde{k}$
$\widetilde{\widetilde{\mathbf{K}}}$	diagonal matrix of $\tilde{\tilde{k}}$
K_1	range of the eigenvalues of $\mathbf{A}$
K_2	fraction between the eigenvalues $\lvert\lambda_{\max}\rvert$ and $\lvert\lambda_{\min}\rvert$ of $\mathbf{A}$
k	wavenumber of the layer
k_x	first part of the decomposition of k
k_y	second part of the decomposition of k
k_1	wavenumber of first layer
k_2	wavenumber of second layer

Symbol	*Meaning*
$\hat{k}$	SoV-wavenumber
$\tilde{k}$	remaining part of the decomposition of k
$\tilde{\tilde{k}}$	modified form of $\tilde{k}$
$\mathbf{L}$	undertriagonal of $\mathbf{A}$
L	length of the computation box
$\mathbf{M}$	preconditioner
$\mathbf{M}_{CSL}$	complex shifted Laplace preconditioner
$\mathbf{M}_{mod(i)}$	modified preconditioner based on separation of variables
$\mathbf{M}_{SL}$	real shifted Laplace preconditioner
$\mathbf{M}_{SoV}$	preconditioner based on separation of variables
M	number of gridpoints in y-direction
m	count coefficient
m	right-angle direction w.r.t. the outward normal n
N	number of gridpoints in x-direction
n	outward normal to a boundary
n	count coefficient
$\mathbf{P}$	permutation matrix
$\mathbf{p}$	pressure vector
p	pressure
$\mathbf{r}_i$	residual $\|\|\mathbf{f} - \mathbf{A}\mathbf{p}_i\|\|_2$
t	time
$\mathbf{v}$	modified $\mathbf{p}$ as in the system $\mathbf{D}\mathbf{v} = \mathbf{g}$
$\mathbf{v}$	eigenvector of $\mathbf{A}$
$\mathbf{v}_i$	eigenvector of $\mathbf{A}$
$\mathbf{W}_R$	right eigenvector matrix
$\mathbf{W}_L$	left eigenvector matrix
W	maximal number of waves
X	length of the x-axis in the domain
$\mathbf{x}$	(unknown) vector of location
x	horizontal direction
Y	length of the y-axis in the domain
y	vertical direction
z	axial direction

Greek Symbols

Symbol	*Meaning*
α	part of x-direction
α	real parameter in SL or CSL preconditioner
β	part of y-direction
β	imaginary parameter in SL or CSL preconditioner
Δ	Laplace operator
δ	delta function
δ_i	relative diffence between two eigenvalues of the HP with S-ABC and DBC
γ	maximal wave amplitude of p
γ	contribution of the boundary to the main diagonal of $\mathbf{A}$
Λ	eigenvalue matrix
λ	wavelength
λ	eigenvalue
ω	wave frequency
Ω	domain, region
σ	number of CSL iterations in modified SoV preconditioner
ϑ	relative residual criterium

Combined Symbols

Symbol	*Meaning*
Δx	grid spacing in x-direction
Δy	grid spacing in y-direction
$\partial\Omega$	boundaries of Ω

Other Symbols

Symbol	*Meaning*
$\mathcal{K}_j$	Krylov subspace with dimension j
$\mathcal{R}$	real part

List of Abbreviations

Abbreviation	*Meaning*
AILU	Analytic ILU preconditioner
AP	Alternated preconditioner
AP–K	Alternated preconditioner using $\tilde{k}(x,y)$
Bi-CG	Biconjugate gradient method
Bi-CGSTAB	Biconjugate gradient stabilized method
C1	Alternative choice 1 of constructing k_x and k_y
C2	Alternative choice 2 of constructing k_x and k_y
CG	Conjugate gradient method
CGS	Conjugate gradient squared method
CS-ABC	Conjugate Sommerfeld absorbing boundary conditions
CSL	Complex shifted Laplace preconditioner
D	Diagonal preconditioner
DBC	Dirichlet boundary conditions
DHP	Discrete Helmholtz problem
GCR	Generalized conjugate residual method
GJ	Gauss-Jacobi (preconditioner)
GS	Gauss-Seidel (preconditioner)
GMRES	Generalized minimum residual method
HP	Helmholtz problem
IC	Incomplete Cholesky preconditioner
ILU	Preconditioner based on incomplete LU-decomposition
MP	Multiplication preconditioner
PSD	Positive semi-definite
PD	Positive definite
RR	Relative residual (criterium)
SL	Shifted Laplace preconditioner
S-ABC	Sommerfeld absorbing boundary conditions
SoV	Preconditioner based on separation of variables
SP	Sum / additive preconditioner

Chapter 1

Introduction

Wave propagation through an inhomogeneous acoustic medium with a constant density is described in the frequency domain by the *Helmholtz equation*:

$$-\Delta p(\mathbf{x}) - k(\mathbf{x})^2 p(\mathbf{x}) = f(\mathbf{x}), \tag{1.1}$$

where p is the wave field and f a (point) source term. The wavenumber $k = \omega/c$ is a function of the spatial coordinates $\mathbf{x} = (x, y)$ in the 2-dimensional case and $\mathbf{x} = (x, y, z)$ in the 3-dimensional case, because the velocity c depends on the spatial coordinates in an inhomogeneous medium.

Background

Equation (1.1) is used for instance for seismic investigations of the earth's crust. The results can be applied to determine the position of various layers. Thereafter, potential locations of oil or gas reservoirs can be predicted.

The Helmholtz equation arises in many physical applications, not only in scattering problems of acoustics but also in scattering problems of electromagnetics, see e.g. Abrahamsson & Kreiss [1]. In Figure 1.1 one can find an application of our interest, where seismic reflections can be modelled with the Helmholtz equation. A pressure wave propagates from the source and is reflected or refracted at discontinuities in the subsurface. The pressure field is recorded at the receiver locations. In general, the modeling requires the computation of the wave propagation in a highly inhomogeneous medium.

Helmholtz Problem

In practice, when modeling a seismic survey, the wavenumber is such that the operator has positive and negative eigenvalues. To mimic an infinite space by a finite computational domain, *absorbing* boundary conditions are added. These boundary conditions make the system invertible, although it remains ill-conditioned, see Mulder & Plessix [37].

In this thesis we consider only the seismic survey, where p in Equation (1.1) is the *pressure* field and absorbing boundary conditions are used. The Helmholtz equation with these boundary conditions leads to the Helmholtz boundary valued problem, or briefly the Helmholtz problem (HP). When a second-order

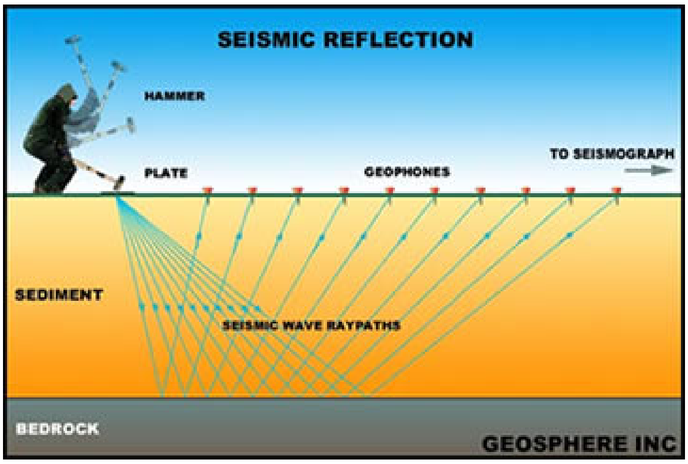


Figure 1.1: Seismic reflection line is a set of seismographs usually lined up along the earth's surface to record seismic waves generated by an explosion for the purpose of recording reflections of these waves from velocity discontinuities within the earth. The data collected can be used to infer the internal structure of the earth. [Source: http://earthquake.usgs.gov/image_glossary]

finite-difference discretization is applied to the HP, we obtain the linear system

$$\mathbf{A}\mathbf{p} = \mathbf{f}, \tag{1.2}$$

where $\mathbf{A}$ is a large but sparse matrix. The solution $\mathbf{p}$ is represented on a grid between 500 and 2000 points per coordinate direction in typical seismic applications.

Solvers

For our discretization of the 2-dimensional version of Equation (1.1), it is necessary that the number of grid points grows faster than quadratically in the wavenumber in order to maintain a given accurary, see Bayliss, Goldstein & Turkel [7]. Thus, for sufficiently high wavenumber, the discretized Helmholtz equation 'leads to a huge linear system of equations', see Abrahamsson & Kreiss [1]. This huge system (1.2) can be solved by a direct method based on LU-factorization. With the nested dissection reordering method using n points in each coordinate direction, the complexity of the LU-factorization is $\mathcal{O}(n^3)$ and the computation of the solution has a cost of $\mathcal{O}(n^2 \log n)$, which is the optimal complexity for direct methods in general, see George & Liu [19].

In the 3-dimensional case, a direct solver is generally too expensive. For instance, the complexity of the LU-factorization with the nested dissection method is $\mathcal{O}(n^6)$, see Mulder & Plessix [37].

Multigrid methods for the solution of linear systems as in (1.2) are known in the literature, see e.g. Goldstein [22]. A common disadvantage of these methods is that the coarsest level must be fine enough to capture the wave character of the problem. Due to that reason, the iterations in our system (1.2) diverge, which is also a direct consequence of the indefiniteness of matrix $\mathbf{A}$.

Moreover, since we focus on the modeling of a seismic survey, the velocity is either a smooth model or a model consisting of a large number of inhomogeneous layers, which is unfavourable for the use of a domain decomposition approach, see also Mulder & Plessix [37].

To fully take into account the sparseness of **A**, an *iterative* method should be applied instead of direct, domain decomposition or multigrid methods. In the literature, the linear system (1.2) is often solved by an iterative preconditioned method based on Krylov spaces due to their relative robustness, for instance by the GMRES method (Saad & Schultz [40]) or by the Bi-CGSTAB method (Van der Vorst [49]).

Moreover, Bayliss, Goldstein & Turkel [6] used a preconditioned conjugate gradient method applied to the normal equations (also known as CGNR). Due to the ill-conditioning of the normal equations, the unpreconditioned algorithm suffered from extremely slow convergence. The convergence rate was substantially improved through preconditioners based on SSOR (also in [6]) or a multigrid V-cycle (Bayliss, Goldstein & Turkel [8]) or only for the Laplacian part of the Helmholtz operator (Goldstein [21]).

As is well known, the convergence rate depends on the spectral properties of the (preconditioned) matrix (see e.g. Golub & Van Loan [23] and Saad [39]). In the case of interest to us, matrices involved are complex-symmetric, indefinite and hard to solve as the frequency, or equivalently, the acoustic wavenumber increases. In such a case, the eigenvalues tend to be scattered between both the right and the left half-plane, see e.g. Ernst & Golub [16]. In order to avoid the convergence of the Krylov methods to dramatically deteriorate, it is of essential importance to use a powerful *preconditioner*.

Preconditioners

Besides the above called preconditioners, there are several preconditioners proposed by e.g. Erlangga, Vuik & Oosterlee [15, 52], Gander & Nataf [18], Laird [29], Made [31] and Plessix & Mulder [37].

Laird [29] constructed the preconditioner based on the Laplace operator perturbed by a real-valued linear term, also known as the shifted Laplace (SL) preconditioner. This suprisingly straightforward idea leads to very satisfactory convergence, where the preconditioning matrix allows the use of SSOR, ILU or multigrid to approximate the inversion within an iteration. Erlangga [15, 52] introduced a complex perturbation to the Laplace operator, which in general results in a better preconditioner than using a real-valued perturbation. We call this the complex shifted Laplace (CSL) preconditioner or briefly 'CSL'. This class of preconditioners is simple to construct and is easily extend to inhomogeneous media.

Gander & Nataf [18] proposed AILU which is a preconditioner based on the analytic factorization of an elliptic operator. An incomplete factorization-based preconditioner is introduced by Made [31]. Numerical experiments in [18, 31] illustrate the effectiveness of these approaches.

Plessix & Mulder [37] have created a preconditioned iterative method based on separation of variables (SoV preconditioner or briefly 'SoV'). For smooth models and low wavenumbers, the convergence rate with this preconditioner is satisfactory. Unfortunately, it rapidly deteriorates when the roughness of the model or the wavenumber increases.

1.1 Objective of the Thesis

In this thesis we concentrate on the preconditioners of Mulder & Plessix (SoV preconditioner) and Erlangga, Oosterlee & Vuik (CSL preconditioner). CSL gives bad convergence in cases when matrix $\mathbf{A}$ in (1.2) has many eigenvalues near zero. It appears that these 'bad' eigenvalues do not disappear in the CSL preconditioned system, leading to the failure of the preconditioner, see also [15, 52].

The SoV preconditioner works well for smooth models and low wavenumbers, thus in situations when the model is 'almost' separable. First we investigate the eigenvalue distribution of the SoV preconditioned system since it is rather unknown in literature. In the most favourable scenario the 'bad' eigenvalues in matrix $\mathbf{A}$ of (1.2) do not correspond with the eigenvalues near zero of the SoV preconditioner. In this ideal case, a combination of CSL and SoV can be very attractive, since CSL gets rid of eigenvalues relatively far from zero and SoV gets rid of bad eigenvalues near zero.

1.2 Outline of the Thesis

First of all, the problem formulation is made concrete. The 2-dimensional Helmholtz Problem (HP) and the resulting linear system as in (1.2) are described in more detail in Chapter 2. Furthermore, test problems with different models for choosing k are considered.

A few theoretical results related to the linear algebra, which are essential in our research project, are given in Chapter 3. They give insight in the properties of preconditioners and in the eigenvalue distribution and corresponding eigenvectors of (preconditioned) sytems used in this thesis.

Subsequently, in Chapter 4 the iterative methods and the preconditioners, which we apply in this research, are briefly considered.

In Erlangga, Vuik & Oosterlee [15, 52], Dirichlet instead of absorbing boundary conditions have been used in the eigenvalue analysis of the CSL preconditioner. In the case that CSL is chosen real-valued, the matrix $\mathbf{A}$ as well the eigenvalues of the (preconditioned) system are also real-valued, while the number of iterations of the iterative methods are more or less equal. In Chapter 5, it can be seen that this observation does not hold for the SoV preconditioned system, since the eigenvalues in this case are still complex, in general.

In Chapter 6, a comparison is made between applying SoV and CSL in iterative methods, by examining their convergence behavior in small test problems using various models for the wavenumber.

Chapter 7 deals with all aspects of the SoV preconditioner using small test problems. This preconditioner is compared with other preconditioner in simple models to emphasize the power of the preconditioner. Mathematical reasons are given for the failure of the SoV preconditioner in rough models or models with high wavenumbers. Moreover, several attempts are made to improve SoV.

Before starting with the combination of the CSL and SoV preconditioners, we investigate whether there are possibilities for creating such a combination using eigenvalue and eigenvector analysis. This is done in Chapter 8.

Chapter 9 deals, finally, with the combinations based on the CSL and SoV. Several variants are treated and the performance of these combined preconditioners (CP) are compared with the original CSL and SoV preconditioners.

We end with Chapters 10 and 11, where the conclusions and some recommendations for further research are drawn.

Chapter 2

Problem Formulation

In this chapter, the problem, which is treated in this thesis, is stated in more detail. The wave equation and the related Helmholtz equation are introduced. This Helmholtz equation with absorbing boundary conditions leads to the Helmholtz Problem (HP). We discretize the HP which results in a linear system.

Furthermore, various layer models are given which we use in the test problems. The other parameters of the test problems are also given at the end of this chapter.

2.1 Wave Equation

The *wave equation* is an important partial differential equation which generally describes all kinds of waves, such as sound waves, light waves and water waves. It arises in many different fields, such as acoustics, electromagnetics and fluid dynamics, see e.g. Achenbach [3] and Colton & Kress [12], where also the derivation of the wave equation can be found.

The general form of the wave equation in *two dimensions* is

$$\frac{\partial^2 p}{\partial t^2}(\mathbf{x},t) = c(\mathbf{x})^2 \Delta p(\mathbf{x},t), \tag{2.1}$$

where $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ (Laplace operator), $\mathbf{x} = (x,y)$ (coordinates/location in two directions in domain Ω) and $t > 0$ (time). Moreover, the speed of the wave propagation c is a function of the location. In seismic applications, which is of our interest, c is in fact the background velocity in the layers of the surface and varies typically between 1500 m/s and 4000 m/s. The amplitude (and in the seismic applications: the pressure wave) is denoted by $p(\mathbf{x},t)$, which is a measure of the intensity of the wave at a particular location $\mathbf{x}$ and time t.

2.2 Helmholtz Equation

We consider *time-harmonic* (standing) waves with *time-independent* pressure $\tilde{p}$. Then $\tilde{p}$ satisfies

$$p(\mathbf{x},t) = \gamma \cos(\omega t)\tilde{p}(\mathbf{x}), \tag{2.2}$$

where $\omega > 0$ is the *wave frequency* and $\gamma > 0$ is the maximum wave amplitude. Since the wavelength λ is equal to $\lambda = \frac{2\pi}{\omega}$ and also equal to $\lambda = \frac{1}{f}$, one obtains $\omega = 2\pi f$, where f is the frequency of the waves which typically varies between 10 Hz and 50 Hz in seismic applications.

Equation (2.2) may also be written as

$$p(\mathbf{x}, t) = \Re\gamma e^{-i\omega t}\tilde{p}(\mathbf{x}), \tag{2.3}$$

where $i^2 = -1$ and $\Re$ indicates that the real part of the right-hand-side of (2.3) should be taken. We shall omit this symbol $\Re$ in further analysis for brevity and keep in mind that we are mainly interested in the real part of complex solutions. However, it appears that in some cases the complex part of the solutions do give some relevant information, so these are not fully neglected in the following of this thesis.

Using Equation (2.3), the wave equation (2.1) reduces to the *Helmholtz equation*:

$$-\Delta\tilde{p}(\mathbf{x}) - k(\mathbf{x})^2\tilde{p}(\mathbf{x}) = 0, \tag{2.4}$$

where the *wave number* k is defined by

$$k(\mathbf{x}) = \frac{\omega}{c(\mathbf{x})}. \tag{2.5}$$

Equation (2.4) carries the name of *Von Helmholtz* for his contributions to mathematical acoustics and electromagnetics. Furthermore, the minus-signs at the left-hand-side of (2.4) have been added due to numerical reasons [1].

***Figure 2.1:* A German stamp of Hermann Ludwig Ferdinand von Helmholtz (1821-1894), one of the greatest physicists and mathematicians.**

If there is no ambiguity in the context, we denote the quantity $\tilde{p}(\mathbf{x})$ by $p(\mathbf{x})$ in this thesis.

[1] Note that $-\Delta$ is a *positive* operator, which means that it has *positive* eigenvalues. Adding an extra term $-k(\mathbf{x})^2$ gives us the operator $-\Delta - k(\mathbf{x})^2$ which is used in (2.4).

Moreover, if we assume an *harmonic source* in the neighbourhood of Ω, i.e., an harmonic disturbance $g(\mathbf{x}) = e^{-i\omega t} f(\mathbf{x})$, which is producing the waves, then the source appears on the right-hand-side of (2.4). As a consequence, we obtain the *inhomogeneous Helmholtz equation*

$$-\Delta p(\mathbf{x}) - k(\mathbf{x})^2 p(\mathbf{x}) = f(\mathbf{x}). \tag{2.6}$$

This equation is the central equation of this thesis.

2.3 Absorbing Boundary Conditions

In applications, the pressure waves propagate in an infinite domain. For numerical reasons, the computations are performed in a bounded domain Ω, which is a rectangular box with sizes $(x_{\min}, x_{\max}) \times (y_{\min}, y_{\max})$, see also Section 2 of Plessix & Mulder [37]. With the help of a *translation* operator, we can easily rewrite the problem to a domain with sizes $(0, \hat{x}_{\max}) \times (0, \hat{y}_{\max})$ where $\hat{x}_{\max} = x_{\max} - x_{\min}$ and $\hat{y}_{\max} = y_{\max} - y_{\min}$.

Appropriate boundary conditions have to be chosen such that the solution at the finite Ω represents the 'real' solution at an infinite domain as well as possible. Several approaches of this problem have been proposed and good summaries of much of the work that has been done on this problem are described by e.g. Givoli [20] and Moore *et al.* [33]. Radiation boundary conditions designed for use on a circular artificial boundary have been introduced by Bayliss & Turkel [9, 10]. On the basis of these is the *Sommerfeld radiation condition* which is defined in *spherical* coordinates (r, ϕ):

$$\lim_{r\to\infty} \sqrt{r}\left(\frac{\partial}{\partial r} - ik(r,\phi)\right) p(r,\phi) = 0, \tag{2.7}$$

as proposed by Sommerfeld [46]. The Sommerfeld radiation condition is also known as:

$$\lim_{r\to\infty} r\left(\frac{\partial}{\partial r} + ik(r,\phi)\right) p(r,\phi) = 0, \tag{2.8}$$

see for instance Ochmann & Mechel [36] and Sladek, Tanaka & Sladek [42]. In both (2.7) and (2.8), r denotes the radius in *spherical* coordinates (r, ϕ). This condition ensures that the scattered field corresponds to a purely outgoing wave at infinity, i.e., it basically ensures that sources radiate waves instead of absorbing or reflecting them.

Now, since we have a numerical finite domain, the radius r is also finite. Assume $r = R$ to be the maximum radius with $R > 0$. In this case, possible boundary conditions to impose are:

$$\frac{\partial p(r,\phi)}{\partial r} - ik(r,\phi)p(r,\phi) = 0, \quad r = R, \tag{2.9}$$

and

$$\frac{\partial p(r,\phi)}{\partial r} + ik(r,\phi)p(r,\phi) = 0, \quad r = R. \tag{2.10}$$

In *Cartesian* coordinates, these conditions can be approximated by

$$\text{(S-ABC)} \qquad \frac{\partial p(\mathbf{x})}{\partial n} - ik(\mathbf{x})p(\mathbf{x}) = 0, \quad \mathbf{x} \in \partial\Omega, \tag{2.11}$$

and

$$\text{(CS-ABC)} \qquad \frac{\partial p(\mathbf{x})}{\partial n} + ik(\mathbf{x})p(\mathbf{x}) = 0, \quad \mathbf{x} \in \partial\Omega, \tag{2.12}$$

where n is the unit outward normal to the boundaries. We call these conditions *first-order Sommerfeld* respectively *conjugate Sommerfeld absorbing boundary conditions* or briefly: S-ABC and CS-ABC. Although these first-order conditions are regular inaccurate in practice (see e.g. Kim [27, 28]), we do take CS-ABC (2.12) as the standard boundary conditions in our problems in this thesis. In future research, we may use other more appropriate boundary conditions to reduce reflections at the boundaries as well as possible.

2.4 Helmholtz Problem

Wave propagation in an *inhomogeneous* medium is considered, which means that the medium consists of several layers. In this thesis we assume that the background velocity c, and therefore also the wavenumber k, is constant in each layer. In other words: an inhomogeneous medium Ω is considered with homogeneous layers.

Let n denote the *unit outward normal* to $\partial\Omega$ where $\partial\Omega$ is the boundaries of Ω. Then, the *continuous Helmholtz's boundary value problem* or briefly the *Helmholtz's problem* (HP) for wave propagation in an *inhomogeneous* medium can be defined as follows:

Helmholtz Problem (HP)
Find the total pressure field $p(\mathbf{x})$ in an inhomogeneous medium Ω such that

$$\left(-\Delta - k(\mathbf{x})^2\right) p(\mathbf{x}) = f(\mathbf{x}), \quad \mathbf{x} \in \Omega, \tag{2.13}$$

with absorbing boundary conditions

$$\left(\frac{\partial}{\partial n} + ik(\mathbf{x})\right) p(\mathbf{x}) = 0, \quad \mathbf{x} \in \partial\Omega, \tag{2.14}$$

where f is a given source term and

$$k(\mathbf{x}) = \frac{w}{c(\mathbf{x})}, \tag{2.15}$$

with wave frequency $\omega > 0$ and background velocity $c > 0$.

The HP can be solved by an integral equation method by transformation into a Fredholm integral equation, see for instance Colton & Kress [12]. In the discretized form, the *Fredholm integral equation* results in a large full matrix which requires the inversion for resolving the solution. This is considered too expensive in many practical problems.

In this thesis, we aim at *numerical* solutions of HP by applying a *finite difference approach*, see the next section.

2.5 Finite Differences

With respect to the finite domain Ω with sizes $(0, \hat{x}_{\max}) \times (0, \hat{y}_{\max})$, one can discretize Ω vertex-centered on a equidistant grid with $M+2$ points in x-direction and $N+2$ points in y-direction, using:

$$\left\{ \begin{array}{lcllcl} x_m & = & x_0 + m\Delta x, & \text{for } m & = & 0, 1, \ldots, M+1; \\ y_n & = & y_0 + n\Delta y, & \text{for } n & = & 0, 1, \ldots, N+1; \end{array} \right. \tag{2.16}$$

with constant parameters $\Delta x, \Delta y > 0$ and where $(x_0, y_0) = (0, 0)$ and $(x_{M+1}, y_{N+1}) = (\hat{x}_{\max}, \hat{y}_{\max})$. Now, the discretization based on *second-order finite differences* [2] leads to

$$-\frac{p_{m-1,n} - 2p_{m,n} + p_{m+1,n}}{\Delta x^2} - \frac{p_{m,n-1} - 2p_{m,n} + p_{m,n+1}}{\Delta y^2} - k_{m,n}^2 p_{m,n} = f_{m,n}, \tag{2.17}$$

for $m = 1, \ldots, M$ and $n = 1, \ldots, N$, with $M, N \in \mathbb{N}$. The notations $p_{m,n} \equiv p(x_m, y_n)$ and $f_{m,n} \equiv f(x_m, y_n)$ are used in (2.17) for simplicity.

The system of linear equations of (2.17) can be written as the linear system

$$\mathbf{A}\mathbf{p} = \mathbf{f}, \tag{2.18}$$

where $\mathbf{p}$ and $\mathbf{f}$ both are vectors with MN elements satisfying

$$\begin{array}{rcl} \mathbf{p}(m + (n-1)N) & = & p_{m,n}; \\ \mathbf{f}(m + (n-1)N) & = & f_{m,n}, \end{array} \tag{2.19}$$

for $m = 1, 2, \ldots, M$ and $n = 1, 2, \ldots, N$. Moreover, $\mathbf{A}$ is a matrix with dimensions $MN \times MN$ and with 5 non-zero diagonals. These diagonals are given by

$$\left\{ \begin{array}{lcl} \mathbf{A}(d, d) & = & \dfrac{2}{\Delta x^2} + \dfrac{2}{\Delta y^2} - k_{m,n}^2 + \gamma_{m,n}; \\ \mathbf{A}(d, d+1) & = & \mathbf{A}(d, d-1) = -\dfrac{1}{\Delta x^2}; \\ \mathbf{A}(d, d+M) & = & \mathbf{A}(d, d-M) = -\dfrac{1}{\Delta y^2}, \end{array} \right. \tag{2.20}$$

for $d = 1, 2, \ldots, MN$. In (2.20) we have

$$\gamma_{m,n} = 0, \quad \text{for } m = 2, 3, \ldots, M-1 \text{ and } n = 2, 3, \ldots, N-1, \tag{2.21}$$

[2] More information about finite difference methods can be found in for instance Mitchell & Griffiths [32].

and at the boundaries:

$$\gamma_{m,n} = \begin{cases} \gamma_x^{\min}(n) & = & \dfrac{1}{\Delta x^2(1 + ik_{0,n}\Delta x)} & \text{if } m = 1; \\[2ex] \gamma_x^{\max}(n) & = & \dfrac{1}{\Delta x^2(1 + ik_{M+1,n}\Delta x)} & \text{if } m = M; \\[2ex] \gamma_y^{\min}(m) & = & \dfrac{1}{\Delta y^2(1 + ik_{m,0}\Delta y)} & \text{if } n = 1; \\[2ex] \gamma_y^{\max}(m) & = & \dfrac{1}{\Delta y^2(1 + ik_{m,N+1}\Delta y)} & \text{if } n = N. \end{cases} \tag{2.22}$$

Note that we have used a first-order scheme to discretize these boundaries, whereas a second-order scheme is used for the interior points of the domain. Now, the Helmholtz problem can be written in the following way.

> **Discretized Helmholtz Problem (DHP)**
> *Let $\boldsymbol{f}$ be a given vector of size MN. Find the vector $\mathbf{p}$ such that*
>
> $$\mathbf{A}\mathbf{p} = \mathbf{f}, \tag{2.23}$$
>
> *where $\mathbf{A}$ is a $MN \times MN$ matrix satisfying Expressions (2.20)–(2.22).*

One can verify that matrix $\mathbf{A}$ has the following properties:

- $\mathbf{A}$ consists of 5 non-zero diagonals, so $\mathbf{A}$ is *sparse*;
- $\mathbf{A}$ is real-valued except the main-diagonal, which is in general complex;
- $\mathbf{A}$ is *complex-symmetric*;
- $\mathbf{A}$ is (strongly) *indefinite* in general, i.e., $\mathbf{A}$ consists of both (large) positive and negative eigenvalues.

2.6 Relation between Wavenumber and Domainsize

It appears that the HP and the DHP can be modified such that it is spectrally equivalent to problems on the unit domain $\widetilde{\Omega}$ ($= (0,1) \times (0,1)$ or briefly $(0,1)^2$), by adapting the wavenumber properly. This can be seen in the following way.

The Helmholtz equation can be written as

$$-\frac{\partial^2}{\partial x^2}p(\mathbf{x}) - \frac{\partial^2}{\partial y^2}p(\mathbf{x}) - k(\mathbf{x})^2 p(\mathbf{x}) = f(\mathbf{x}), \tag{2.24}$$

where $\mathbf{x} = (x, y) \in \Omega$. We take $\Omega = (0, L)^2, L \in \mathbb{R}$ for simplicity.

Next, we introduce the following new variables:

$$\begin{cases} \tilde{x} = \dfrac{x}{L}, \\[2ex] \tilde{y} = \dfrac{y}{L}. \end{cases} \tag{2.25}$$

Since $\mathrm{d}\tilde{x}/\mathrm{d}x = \mathrm{d}\tilde{y}/\mathrm{d}y = 1/L$, Equation (2.24) can be written as

$$-\frac{\partial^2}{\partial\tilde{x}^2}p(\tilde{\mathbf{x}}) - \frac{\partial^2}{\partial\tilde{y}^2}p(\tilde{\mathbf{x}}) - k^2L^2p(\tilde{\mathbf{x}}) = L^2f(\tilde{\mathbf{x}}), \tag{2.26}$$

with $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y})$ and $(\tilde{x}, \tilde{y}) \in \widetilde{\Omega}$. Then it yields

$$-\frac{\partial^2}{\partial\tilde{x}^2}p(\tilde{\mathbf{x}}) - \frac{\partial^2}{\partial\tilde{y}^2}p(\tilde{\mathbf{x}}) - \hat{k}^2p(\tilde{\mathbf{x}}) = \hat{f}(\tilde{\mathbf{x}}), \tag{2.27}$$

where

$$\left\{ \begin{array}{rcl} \hat{k}(\tilde{\mathbf{x}}) & = & Lk(\mathbf{x}), \\ \hat{f}(\tilde{\mathbf{x}}) & = & L^2f(\mathbf{x}). \end{array} \right. \tag{2.28}$$

In the above procedure we have made the domain Ω '*dimensionless*'. If one compares (2.24) with (2.27), it can be noticed that the domain is changed from $[0, L]^2$ to $[0, 1]^2$, but the solution is the same due to the new variables $\hat{k}^2$ and $\hat{f}$.

Moreover, since the eigenvalues of (2.24) are independent of the source term $f(\mathbf{x})$, the change of the original domain to the unit domain influences in fact only wavenumber k in spectral analysis. Indeed, the (D)HP is spectral equivalent to the same problem at unit domain up to k.

Example

Assume we use the following variables:

$$(\mathrm{I}) = \left\{ \begin{array}{rcl} k_1^2 & = & 1; \\ k_2^2 & = & 3; \\ \Omega & = & (0, 20)^2 \quad (\text{i.e., } L = 20), \end{array} \right. \tag{2.29}$$

in test problems. Using the theory above, this problem is spectrally equivalent to

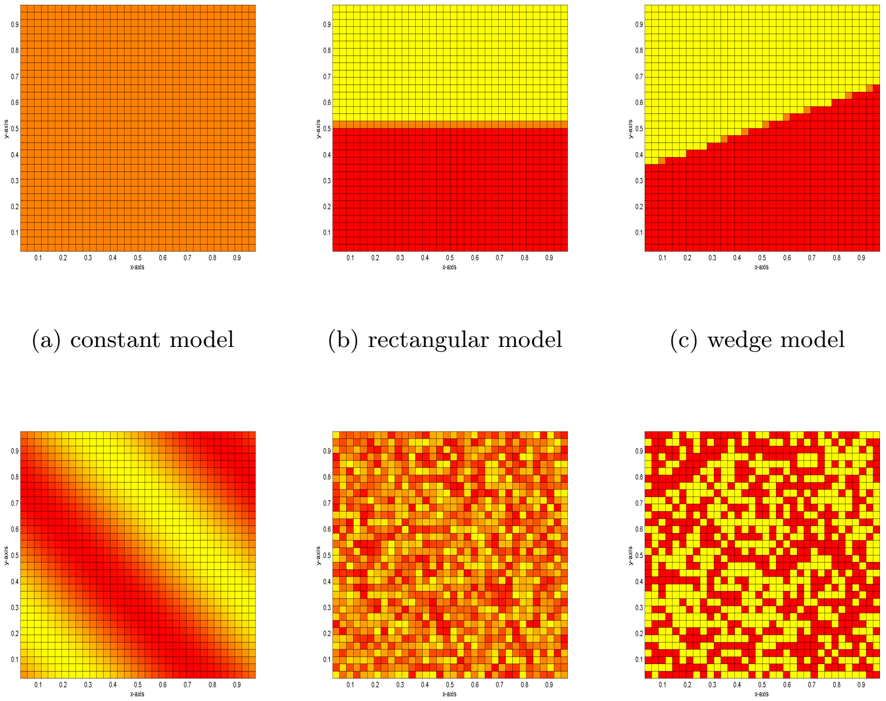
$$(\mathrm{II}) = \left\{ \begin{array}{rcl} \hat{k}_1^2 & = & (k_1L)^2 = 400; \\ \hat{k}_2^2 & = & (k_2L)^2 = 1200; \\ \Omega & = & (0, 1)^2. \end{array} \right. \tag{2.30}$$

2.7 Layer Models

There are many (realistic) layer models available which model the wavenumber k in Ω. In this thesis we consider a few simple models: the constant, rectangular, wedge, sinus, random and min-max models, which are described briefly below and which can also be seen in Figure 2.2. The central one in our test problems is the *wedge model* where special attention is given below, also to its numerical implementation.

Constant Model

In the constant model, the wavenumber k is chosen to be fully constant in the whole domain.



(a) constant model (b) rectangular model (c) wedge model

(d) sinus model (e) random model (f) min-max model

Figure 2.2: Plots of the numerical implementation of all layer models which are used in test problems of this thesis. In these plots the wavenumber is depicted on the unit domain after discretization.

Rectangular Model

The domain consists of two 'rectangular' layers with wavenumber k_1 and k_2, respectively, where the (common) interface is horizontally located. Furthermore, we assume that $k^2 = \frac{k_1^2+k_2^2}{2}$ holds at this interface.

Wedge Model

The wedge model [3] is almost the same as the layer model, but now the layers have the form of a trapezium instead of a rectangle, i.e., the layers are separated by a diagonal interface. Assume that this diagonal interface starts (in $x = 0$) at αY and ends (in $x = X$) at βY, where $0 < \alpha \leq \beta < 1$. The geometry can be found in Figure 2.3.

The direction coefficient of this diagonal interface is exactly

$$\frac{\Delta y}{\Delta x} = \frac{(\beta - \alpha)Y}{X}, \tag{2.31}$$

[3] The wedge model is actually a model with three layers as in Plessix & Mulder [37], but for simplicity we take only two layers in this thesis.

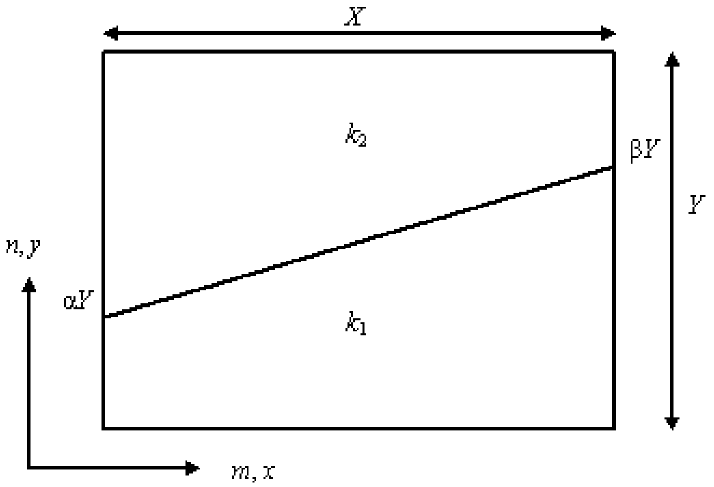


Figure 2.3: Geometry of the wedge model.

which results in the following equation of the diagonal interface:

$$y = \frac{(\beta - \alpha)Y}{X}x + \alpha Y, \quad x \in [0, X]. \tag{2.32}$$

Now, we can say that if

$$y < \frac{(\beta - \alpha)Y}{X}x + \alpha Y, \quad x \in [0, X], \tag{2.33}$$

then we deal with the first layer where $k = k_1$. In the case of

$$y > \frac{(\beta - \alpha)Y}{X}x + \alpha Y, \quad x \in [0, X], \tag{2.34}$$

then we have $k = k_2$, since this is in the region of the second layer. Finally, if we obtain exactly

$$y = \frac{(\beta - \alpha)Y}{X}x + \alpha Y, \quad x \in [0, X], \tag{2.35}$$

then we are in the middle of the layers and the corresponding wavenumber is assumed to be equal to $\frac{k_1^2+k_2^2}{2}$.

Next, the numerical implementation of the wedge model, using gridelements, can be done in the following way. Divide Ω into elements (m, n) such that horizontally and vertically we obtain exactly M and N pieces, respectively, thus: $m = 1, 2, \ldots M$ and $n = 1, 2, \ldots, N$. Now, we need to find a relation between M, N on the one hand and x, y on the other hand, before we are able to discretize our problem.

First, we find a linear relation of y at n. Two couples of (n, y) are for instance $\left(1, 0 + \frac{Y}{N+1}\right)$ and $\left(N, Y - \frac{Y}{N+1}\right)$, which gives us immediately the relation:

$$y = \frac{Y}{N+1}n, \quad n \in [0, N]. \tag{2.36}$$

In the same way, we obtain

$$x = \frac{X}{M+1} m, \quad m \in [0, M], \tag{2.37}$$

where we have used the fact that $\left(1, 0 + \frac{X}{M+1}\right)$ and $\left(M, X - \frac{X}{M+1}\right)$ are two couples of the set (m, x).

Now we substitute the relations (2.36) and (2.37) into Equation (2.32), resulting in:

$$n = (N+1)\left(\frac{(\beta - \alpha)m}{M+1} + \alpha\right). \tag{2.38}$$

Hence, we can distinguish three cases:

$$k^2 = \begin{cases} k_1^2, & \text{if } n < (N+1)\left(\dfrac{(\beta-\alpha)m}{M+1} + \alpha\right); \\ \dfrac{k_1^2 + k_2^2}{2}, & \text{if } n = (N+1)\left(\dfrac{(\beta-\alpha)m}{M+1} + \alpha\right); \\ k_2^2, & \text{if } n > (N+1)\left(\dfrac{(\beta-\alpha)m}{M+1} + \alpha\right). \end{cases} \tag{2.39}$$

Notice that if we take $\alpha = \beta = \frac{1}{2}$, we obtain exactly the rectangular model.

Sinus Model

The sinus model considers a wavenumber k such that

$$k^2(x, y) = A + B \sin(2\pi(x + y)), \tag{2.40}$$

where

$$A = \frac{k_1^2 + k_2^2}{2}, \quad B = \frac{|k_2^2 - k_1^2|}{2}, \tag{2.41}$$

with k_1 and k_2 to be constants. In other words, the wavenumber k^2 is modelled as a sinusoide with equilibrium position A and amplitude B. As a consequence, k varies between k_1 and k_2.

Random Model

The random model is defined in the following way:

$$k^2(x, y) = k_1^2 + \chi(x, y) \cdot |k_1^2 - k_2^2|, \tag{2.42}$$

where $\chi(x, y)$ is a random non-differentiable real function in the range of $[0, 1]$. In this case k varies exactly between k_1 and k_2.

Min-Max Model

The so-called min-max model takes only two possible wavenumbers in the domain, i.e., the min-max model is defined in the following way:

$$k^2(x,y) = \begin{cases} k_1^2, & \text{if } \chi(x,y) > \frac{1}{2}, \\ k_2^2, & \text{if } \chi(x,y) \leq \frac{1}{2}, \end{cases} \tag{2.43}$$

where $\chi(x,y)$ is again a random non-differentiable real function in the range of $[0,1]$.

2.8 Parameters

The continuous and discretized HP have been formulated in the previous sections. In our test problems, we have to define also the source term $f(x)$, the wavenumber k, the domain lengths $\hat{x}_{\max}$ and $\hat{y}_{\max}$ of the computational box of Ω and the number of gridpoints M and N.

2.8.1 Source Term and Boundary Conditions

In seismic applications of our interest, the pressure field is modelled in the case when there is a kind of explosion somewhere (at location $\mathbf{x}_0$) at the surface. In this case, $f(\mathbf{x})$ is a weighted delta function:

$$f(\mathbf{x}) = \gamma \cdot \delta(\mathbf{x} - \mathbf{x}_0), \;\; \gamma \in \mathbb{R}. \tag{2.44}$$

In fact, $f(\mathbf{x})$ is a point source term. As a consequence, the vector $\mathbf{f}$ consists of zeros except for one element, which is equal to γ.

Furthermore, we have seen earlier in Section 2.3 that the finite domain Ω is taken as a rectangular box $[0, \hat{x}_{\max}] \times [0, \hat{y}_{\max}]$. In this thesis we use

$$\hat{x}_{\max} = \hat{y}_{\max} = L, \;\; L > 0. \tag{2.45}$$

Absorbing conditions hold at the boundaries, except for the boundary at the surface which is considered to be a Dirichlet boundary.

However, in each test problem of this thesis we take for simplicity the explosion in the *middle* of the rectangular box and moreover we take *uniform* boundaries, i.e., $\mathbf{x}_0$ is centrally located in the box and the boundaries are considered to be all absorbing boundaries. Only in Chapter 5, we take test problems with Dirichlet conditions. The aim of that chapter is to give a comparison between several methods and eigenvalue distribution with different choices of boundary conditions.

2.8.2 Gridsizes

In MATLAB, we implement the test problems with at most 45 gridpoints in each direction, leading to a matrix $\mathbf{A}$ with maximum dimensions 3025×3025. Our computer (ATHLON 2000+, 128 MB) was often unable to compute with more gridpoints in numerical experiments, due to the computational time and the lack of memory. Thus, we choose M and N such that $M, N \leq 45$. In future research we can apply larger gridsizes with the aid of available workstations or using other more efficient computer packages like FORTRAN or C++ instead of MATLAB.

2.8.3 Wavenumber

As earlier mentioned, the background velocity c typically varies between 1500 m/s and 4000 m/s and the frequency $f(\mathbf{x})$ between 10 Hz and 50 Hz in realistic situations . This means that the wavenumber $k(\mathbf{x})$ varies between 0.016 and 0.21 m^{-1}, see also Plessix & Mulder [37]. In this situation the waves propagate in an infinite domain. If we restrict this domain into $\Omega = (0, 1000)^2$ m (thus $L = 1000$ m), then we can represent this problem to a unit domain $\Omega = (0, 1)^2$ in the following way:

$$\left\{ \begin{array}{lllllllll} \hat{k}^2_{\text{min}} & = & (k_{\text{min}}L)^2 & = & (0.016 \times 1000)^2 & = & 16^2 & = & 256 \text{ m}^{-1}; \\ \hat{k}^2_{\text{max}} & = & (k_{\text{max}}L)^2 & = & (0.021 \times 1000)^2 & = & 21^2 & = & 441 \text{ m}^{-1}. \end{array} \right. \tag{2.46}$$

where Expressions (2.28) have been used.

For simplicity, we choose equal gridsizes in each direction, i.e., $M = N$ and moreover we require at least 15 gridpoints per wavelength λ to keep an accurate numerical solution. Below, we compute the maximum wavenumber, where the units of the quantities are omitted.

The maximum number of waves W_{max} turns out to be

$$W_{\text{max}} = \frac{M}{G} = \frac{N}{G} = \frac{45}{15} = 3, \tag{2.47}$$

where G is the minimum number of gridpoints per wavelength which we have chosen to be 15 gridpoints.

Subsequently, the minimum wavelength λ_{min} reads

$$\lambda_{\text{min}} = \frac{L}{W_{\text{max}}} = \frac{1}{3}, \tag{2.48}$$

resulting in the maximum wavenumber:

$$k_{\text{max}} = \frac{2\pi}{\lambda_{\text{min}}} = 6\pi. \tag{2.49}$$

In other words, k^2 can be at most

$$k^2_{\text{max}} < 36\pi^2 = 355.3. \tag{2.50}$$

Thus, assuming $M, N \leq 45$, we have to choose k such that $256 \leq k \leq 355.3$ to maintain an *accurate* and *realistic* solution at unit domain.

However, it appears in numerical experiments that test problems with $k \leq 355.3$ m^{-1} are too easy for further analysis. Fortunately, in spectral analysis one can choose k larger ($k > 355.3$), where the gridsizes is kept to be maximum 45 (thus $M, N \leq 45$). While the solution is not accurate anymore, it appears that the spectral analysis still does make sense. [4] The latter statement is further motivated in Chapter 6.

2.9 Test Problems

After defining various layer models and choosing suitable parameters for the HP, test problems are constructed, which are used in this thesis.

The test problems are carried out on *unit domain*, i.e., $\Omega = (0,1)^2$. Furthermore, the most layer models apply the parameters k_1 and k_2. These are the main parameters which are varied in an arbitrary layer model of the test problems, see Table 2.1.

Name	Abbreviation	k_1^2	k_2^2
constant	(C)	300	300
unrealistic constant	(C+)	400	400
very unrealistic constant	(C++)	1200	1200
realistic	(R)	260	350
less realistic	(R+)	100	350
virtual	(V)	400	1200
more virtual	(V+)	1200	1600
much more virtual	(V++)	1200	2400
extra	(E)	400	4000

Table 2.1: Test problems with different values of the wavenumber k which can be chosen in each layer model.

Only in the cases (C) and (R), the resulting test problems are realistic and give accurate solutions when sufficient large gridsizes are used ($M, N \approx 45$ or larger). However, as earlier mentioned, it appears that the other test problems defined in Table 2.1 and also (C) and (R) with relatively small gridsizes [5] do also give some relevant information, considering for instance the iterative behavior and eigenvalue distributions.

[4] In practice, we can see that the iterative methods converge better when the stepsize is small enough for good accuracy. However, when we investigate combined preconditioners, this is not a drawback.

[5] In this thesis we make mainly use of the gridsizes $M, N = 15, 25, 35, 45$ in the defined test problems of Table 2.1.

Chapter 3

Some Theoretical Results

Before treating methods to solve the Helmholtz problem, we deal with some theoretical aspects. These are mainly related to the *linear algebra* and they play a crucial role in the theory of the SoV preconditioner and in the analysis of the eigenvalue distribution of preconditioners.

We start with the Kronecker product, which is used to simplify the notation of matrices with a special structure. Thereafter, we deal with definitions and relations of right and left eigenvectors. Properties about these eigenvectors and their corresponding eigenvalues are given for symmetric, Hermitian and complex-symmetric matrices. These properties are important, since in the Helmholtz problem we deal with a complex-symmetric matrix. In the case of Dirichlet instead of absorbing conditions, we deal with symmetric matrices.

Eigenvalue and eigenvector properties of some special systems are considered at the end of this chapter, which are useful in the spectral analysis of combined preconditioners.

3.1 Kronecker Product

The Kronecker product is a binary matrix operator that maps two arbitrarily dimensioned matrices into a larger matrix with special block structure. Let the $m \times n$ matrix $\mathbf{A}$ and the $p \times q$ matrix $\mathbf{B}$ be given as follows:

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} b_{1,1} & \cdots & b_{1,n} \\ \vdots & \ddots & \vdots \\ b_{m,1} & \cdots & b_{m,n} \end{bmatrix}. \tag{3.1}$$

Then, their *Kronecker product*, denoted $\mathbf{A} \otimes \mathbf{B}$, is the $mp \times nq$ matrix with the block structure

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{1,1}\mathbf{B} & \cdots & a_{1,n}\mathbf{B} \\ \vdots & \ddots & \vdots \\ a_{m,1}\mathbf{B} & \cdots & a_{m,n}\mathbf{B} \end{bmatrix}. \tag{3.2}$$

Example

Let $\mathbf{A}$ and $\mathbf{B}$ be a 2×2 respectively 3×3 matrix such that

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 0 & -1 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}. \tag{3.3}$$

Then, the Kronecker product $\mathbf{A} \otimes \mathbf{B}$ is

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} 1 & 2 & 3 & 2 & 4 & 6 \\ 4 & 5 & 6 & 8 & 10 & 12 \\ 0 & 0 & 0 & -1 & -2 & -3 \\ 0 & 0 & 0 & -4 & -5 & -6 \end{bmatrix}. \tag{3.4}$$

□

3.1.1 Properties

Below, some properties about the Kronecker product are given, where $\mathbf{A}, \mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ are matrices with size $n \times n$.

- The Kronecker product is a bi-linear operator. Given $\alpha \in \mathbb{R}$,

$$\begin{aligned} \mathbf{A} \otimes (\alpha \mathbf{B}) &= \alpha(\mathbf{A} \otimes \mathbf{B}); \\ (\alpha \mathbf{A}) \otimes \mathbf{B} &= \alpha(\mathbf{A} \otimes \mathbf{B}). \end{aligned} \tag{3.5}$$

- The Kronecker product distributes over addition:

$$\begin{aligned} (\mathbf{A} + \mathbf{B}) \otimes \mathbf{C} &= (\mathbf{A} + \mathbf{C}) \otimes (\mathbf{B} + \mathbf{C}); \\ \mathbf{A} \otimes (\mathbf{B} + \mathbf{C}) &= (\mathbf{A} + \mathbf{B}) \otimes (\mathbf{A} + \mathbf{C}). \end{aligned} \tag{3.6}$$

- The Kronecker product is associative, meaning

$$(\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}). \tag{3.7}$$

- The Kronecker product is *not* commutative in general, i.e., usually

$$\mathbf{A} \otimes \mathbf{B} \neq \mathbf{B} \otimes \mathbf{A}. \tag{3.8}$$

- Transpose distributes over the Kronecker product, i.e.,

$$(\mathbf{A} \otimes \mathbf{B})^T = \mathbf{A}^T \otimes \mathbf{B}^T. \tag{3.9}$$

  Note that we can not inverse the orders of (3.9) due to (3.8).

- Matrix multiplication with the Kronecker product can be done in the following way:

$$(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}. \tag{3.10}$$

- When $\mathbf{A}$ and $\mathbf{B}$ are full rank, it yields

$$(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}. \tag{3.11}$$

The proof of these properties and a deeper treatment of the Kronecker product can be found in Graham [24].

3.2 Left and Right Eigenvectors for Real Matrices

In the SoV preconditioner, the left and right eigenvectors are essential components. Therefore, these are worked out in more detail. We assume first $\mathbf{A}$ to be a *real* and square matrix in this section. In the next section the *complex* and square matrix $\mathbf{A}$, which is of our main interest, is considered.

3.2.1 Right Eigenvectors

Let $\mathbf{A}$ be a real $n \times n$ matrix. A vector $\mathbf{v}_R$ of length n satisfying

$$\mathbf{A}\mathbf{v}_R = \lambda_R \mathbf{v}_R, \tag{3.12}$$

is called a *right eigenvector* for $\mathbf{A}$ corresponding to the *right eigenvalue* $\lambda_R \in \mathbb{C}$. Note that if $\mathbf{v}_R$ is a right eigenvector, then $c \cdot \mathbf{v}_R, c \in \mathbb{C}\backslash\{0\}$ is also a right eigenvector. If all eigenvalues are also distinct, then the corresponding right eigenvector to any right eigenvalue λ_R is, except for a scaling factor c, uniquely determined. Furthermore, when λ_R and $\mathbf{A}$ are both real, then it is known that $\mathbf{v}_R$ can also be chosen real (see e.g. Lay [30]).

If $(\lambda_R)_1, (\lambda_R)_2, \ldots, (\lambda_R)_r$ are all eigenvalues and $(\mathbf{v}_R)_1, (\mathbf{v}_R)_2, \ldots, (\mathbf{v}_R)_r$ are the corresponding right eigenvectors, then it is easy to see that the set of right eigenvectors forms a basis of a vector space. If this vector space is of dimension n, i.e., $r = n$, then we can construct an $n \times n$ matrix $\mathbf{W}_R$ whose columns are the right eigenvectors, which has the property that

$$\mathbf{A}\mathbf{W}_R = \mathbf{W}_R \Lambda_R, \tag{3.13}$$

where Λ_R is the $n \times n$ diagonal matrix with the elements $(\lambda_R)_1, (\lambda_R)_1, \ldots,$ $(\lambda_R)_n$. Observe that we can scale each column of $\mathbf{W}_R$, since each eigenvector is determined up to a scaling factor.

3.2.2 Left Eigenvectors

A vector $\mathbf{v}_L$ of length n with the property

$$\mathbf{v}_L^T \mathbf{A} = \lambda_L \mathbf{v}_L^T, \tag{3.14}$$

is called a *left eigenvector* for $\mathbf{A}$ corresponding to the *left eigenvalue* $\lambda_L \in \mathbb{C}$. Similar to the right eigenvectors, if $(\lambda_L)_1, (\lambda_L)_2, \ldots, (\lambda_L)_r$ are all eigenvalues and $(\mathbf{v}_L)_1, (\mathbf{v}_L)_2, \ldots, (\mathbf{v}_L)_r$ are the corresponding left eigenvectors of $\mathbf{A}$, then again the set of left eigenvectors forms a basis of a vector space. If this vector space is of dimension n, then we can construct an $n \times n$ matrix $\mathbf{W}_L^T$ whose columns are the components of the left eigenvectors, which has the property that

$$\mathbf{W}_L^T \mathbf{A} = \Lambda_L \mathbf{W}_L^T, \tag{3.15}$$

where Λ_L is the $n \times n$ diagonal matrix with $(\lambda_L)_1, (\lambda_L)_1, \ldots, (\lambda_L)_n$.

3.2.3 Relation Left and Right Eigenvectors

It is easy to choose the left eigenvectors $(\mathbf{v}_L)_1, (\mathbf{v}_L)_2, \ldots, (\mathbf{v}_L)_n$ and right eigenvectors $(\mathbf{v}_R)_1, (\mathbf{v}_R)_2, \ldots, (\mathbf{v}_R)_n$ so that

$$(\mathbf{v}_L^T)_i \cdot (\mathbf{v}_R)_j = \begin{cases} 1 & \text{if } i = j; \\ 0 & \text{otherwise,} \end{cases} \tag{3.16}$$

or in other words:

$$\mathbf{W}_L^T \cdot \mathbf{W}_R = \mathbf{I}, \tag{3.17}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix. This latter statement is proved in Theorem 3.1. In this theorem, we use the term (right) diagonalizable: $\mathbf{A}$ is (right) *diagonalizable*, if there exists an invertible $n \times n$ matrix $\mathbf{P}$ such that $\mathbf{P}^{-1}\mathbf{AP}$ is equal to a diagonal matrix $\mathbf{D}$.

Theorem 3.1 *Let $\boldsymbol{A}$ be a real-valued diagonalizable matrix. Then $\boldsymbol{W}_L$ and $\boldsymbol{W}_R$ can be constructed such that $\boldsymbol{W}_L^T \cdot \boldsymbol{W}_R = \boldsymbol{I}$.*

Proof. Since the set of right eigenvectors can be chosen such that this set forms a basis of a vector space with the same dimensions of $\mathbf{A}$ (see Lemma 3.1), the inverse of matrix $\mathbf{W}_R$ exists.

Choose now $\mathbf{W}_L^T = \mathbf{W}_R^{-1}$, then we obtain immediately $\mathbf{W}_L^T \cdot \mathbf{W}_R = \mathbf{I}$. The remaining part to show is that $\mathbf{W}_R^{-1}$ consists of left eigenvectors of $\mathbf{A}$.

By definition,

$$\mathbf{AW}_R = \mathbf{W}_R \Lambda_R. \tag{3.18}$$

Multiplying both sides on the left by $\mathbf{W}_R^{-1}$, it yields

$$\mathbf{W}_R^{-1}\mathbf{AW}_R = \Lambda_R, \tag{3.19}$$

after which multiplying on the right by $\mathbf{W}_R^{-1}$, we have

$$\mathbf{W}_R^{-1}\mathbf{A} = \Lambda_R \mathbf{W}_R^{-1}, \tag{3.20}$$

which implies that any row of $\mathbf{W}_R^{-1}$ satisfies the properties of a left eigenvector.

□

Theorem 3.1 has used part (i) of the following lemma.

Lemma 3.1 *Let $\boldsymbol{A}$ be an $n \times n$ matrix.*

(i) $\boldsymbol{A}$ is diagonalizable if and only if it has n linearly independent right eigenvectors.

(ii) If $\boldsymbol{A}$ is diagonalizable with $\boldsymbol{P}^{-1}\boldsymbol{AP} = \boldsymbol{D}$, then the columns of $\boldsymbol{P}$ are the right eigenvectors of $\boldsymbol{A}$ and the diagonal entries of $\boldsymbol{D}$ are the corresponding eigenvalues.

Proof. The proof is straightforward and can be found in, for instance, Nakos & Joyner [34].

□

In the proof of Theorem 3.1 we have seen that choosing $\mathbf{W}_L^T = \mathbf{W}_R^{-1}$ results in $\mathbf{W}_L^T \cdot \mathbf{W}_R = \mathbf{I}$. Note that this is *independent* of the structure of $\mathbf{A}$. Moreover, if $\mathbf{A}$ has special properties like symmetry, then $\mathbf{W}_L^T$ can be more easily computed, see the next subsection.

3.2.4 Properties of Symmetric Matrix

In this subsection, we assume $\mathbf{A}$ to be *symmetric*, i.e., $\mathbf{A}^T = \mathbf{A}$. First a few simple lemma's are considered, follows by some theorems about $\mathbf{W}_R$ and $\mathbf{W}_L$.

Lemma 3.2 *Let $\boldsymbol{A}$ be a symmetric and invertible matrix. Then $\boldsymbol{A}^{-1}$ is also symmetric.*

Proof. We have

$$\left(\mathbf{A}^{-1}\right)^T = \left(\mathbf{A}^T\right)^{-1} = \mathbf{A}^{-1}, \tag{3.21}$$

where we have used Theorem 9 of Nakos & Joyner [34] in the first equality and $\mathbf{A} = \mathbf{A}^T$ in the second equality. This implies $\mathbf{A}^{-1}$ to be symmetric.

□

Lemma 3.3 *Let $\boldsymbol{A}$ be a real-valued symmetric matrix. Then:*

(a) the eigenvalues of $\boldsymbol{A}$ are real;

(b) the eigenvectors corresponding to distinct eigenvalues of $\boldsymbol{A}$ are orthogonal.

Proof of (a). Let λ denote one of the eigenvalues of $\mathbf{A}$, with corresponding right eigenvector $\mathbf{v}$. If λ is complex, then the components of $\mathbf{v}$ will be complex as well. Since $\mathbf{A}$ is real and symmetric, we obtain $\mathbf{A} = \overline{\mathbf{A}} = \mathbf{A}^T$, where $\overline{\mathbf{A}}$ denotes the conjugate of $\mathbf{A}$. [1]

Given that λ is an eigenvalue and $\mathbf{v}$ is a corresponding right eigenvector of $\mathbf{A}$, it yields:

$$\overline{\mathbf{v}}^T\mathbf{A}\mathbf{v} = \overline{\mathbf{v}}^T\lambda\mathbf{v} = \lambda\mathbf{v}^H\mathbf{v} = \lambda||\mathbf{v}||^2. \tag{3.22}$$

Taking the Hermitian of the left-hand-side of the above equation, we obtain

$$\left(\overline{\mathbf{v}}^T\mathbf{A}\mathbf{v}\right)^H = \overline{\left(\overline{\mathbf{v}}^T\mathbf{A}\mathbf{v}\right)^T} = \overline{\left(\mathbf{v}^T\mathbf{A}\overline{\mathbf{v}}\right)} = \overline{\mathbf{v}}^T\mathbf{A}\mathbf{v}. \tag{3.23}$$

Now doing the same with the right-hand side of (3.22) results in

$$\left(\lambda||\mathbf{v}||^2\right)^H = \left(\overline{\lambda||\mathbf{v}||^2}\right)^T = \overline{\lambda}||\mathbf{v}||^2. \tag{3.24}$$

Thus, combining (3.22), (3.23) and (3.24) we see that

$$\overline{\mathbf{v}}^T\mathbf{A}\mathbf{v} = \lambda||\mathbf{v}||^2 = \overline{\lambda}||\mathbf{v}||^2, \tag{3.25}$$

leading to

$$(\lambda - \overline{\lambda})||\mathbf{v}||^2 = 0. \tag{3.26}$$

[1] For example: let

$$\mathbf{A} = \begin{bmatrix} 2-i & 2 \\ 3 & 1+3i \end{bmatrix},$$

then the conjugate of $\mathbf{A}$ is equal to

$$\overline{\mathbf{A}} = \begin{bmatrix} 2+i & 2 \\ 3 & 1-3i \end{bmatrix}.$$

Since $\mathbf{v}$ is an eigenvector, we know that $||\mathbf{v}||^2 \neq 0$. Thus, Equation (3.26) can only be satisfied is when $\lambda - \overline{\lambda} = 0$, which results in

$$\lambda = \overline{\lambda}. \tag{3.27}$$

We conclude that λ is real.

□

Proof of (b). Let λ_1 and λ_2 be two arbitrary distinct eigenvalues of a real symmetric matrix $\mathbf{A}$, with corresponding eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively. Hence,

$$\mathbf{v}_1^T\mathbf{A}\mathbf{v}_2 = \lambda_2\mathbf{v}_1^T\mathbf{v}_2, \quad \mathbf{v}_2^T\mathbf{A}\mathbf{v}_1 = \lambda_1\mathbf{v}_2^T\mathbf{v}_1. \tag{3.28}$$

Now, taking the transpose of the second equation:

$$\left(\mathbf{v}_2^T\mathbf{A}\mathbf{v}_1\right)^T = \left(\lambda_1\mathbf{v}_2^T\mathbf{v}_1\right)^T \quad \rightarrow \quad \mathbf{v}_1^T\mathbf{A}\mathbf{v}_2 = \lambda_1\mathbf{v}_1^T\mathbf{v}_2. \tag{3.29}$$

Comparing this with the Equation (3.28) leads to

$$\mathbf{v}_1^T\mathbf{A}\mathbf{v}_2 = \lambda_1\mathbf{v}_1^T\mathbf{v}_2 = \lambda_2\mathbf{v}_1^T\mathbf{v}_2. \tag{3.30}$$

But since λ_1 and λ_2 are distinct, we know that $\lambda_1 - \lambda_2 \neq 0$. Hence, $\mathbf{v}_1^T\mathbf{v}_2 = 0$, which means that $\mathbf{v}_1$ and $\mathbf{v}_2$ are orthogonal.

□

The following theorem shows how the left eigenvectors can be computed easily, when the right eigenvectors are known. It appears that we are always able to choose $\mathbf{W}_L = \mathbf{W}_R$, if $\mathbf{W}_R$ exists. Moreover, we prove also that the property $\mathbf{W}_L^H\mathbf{W}_R = \mathbf{I}$ is preserved by this choice (after a possible scaling of eigenvectors).

Theorem 3.2 *Let $\boldsymbol{A}$ be a real-valued symmetric matrix.*

(a) *Let $\boldsymbol{W}_R$ be a corresponding right eigenvector matrix. Then a possible left eigenvector matrix is $\boldsymbol{W}_L = \boldsymbol{W}_R$;*

(b) *Let $\boldsymbol{A}$ to have distinct eigenvalues. Then there exists a right eigenvector matrix $\boldsymbol{W}_R$ and a left eigenvector matrix $\boldsymbol{W}_L = \boldsymbol{W}_R$ such that these satisfy $\boldsymbol{W}_L^T\boldsymbol{W}_R = \boldsymbol{I}$.*

Proof of (a). The following expressions are equivalent

$$\begin{aligned}
\mathbf{A}\mathbf{W}_R &= \mathbf{W}_R\Lambda_R; \\
(\mathbf{A}\mathbf{W}_R)^T &= (\mathbf{W}_R\Lambda_R)^T; \\
\mathbf{W}_R^T\mathbf{A}^T &= \Lambda_R^T\mathbf{W}_R^T; \\
\mathbf{W}_R^T\mathbf{A} &= \Lambda_R\mathbf{W}_R^T.
\end{aligned} \tag{3.31}$$

Comparing the first and last step, we obtain $\mathbf{W}_L^T = \mathbf{W}_R^T$ implying $\mathbf{W}_L = \mathbf{W}_R$.

□

Proof of (b). Below we show that $\mathbf{W}_R^T\mathbf{W}_R = \mathbf{I}$ can be constructed. Then the statement $\mathbf{W}_L^T\mathbf{W}_R = \mathbf{I}$ holds, because $\mathbf{W}_L = \mathbf{W}_R$.

If $\mathbf{W}_R^T\mathbf{W}_R = \mathbf{I}$, then $\mathbf{W}_R$ has to be an orthogonal matrix. This orthogonal $\mathbf{W}_R$ can indeed be constructed, which can be shown in the following way.

Denote $\mathbf{W}_R = [\mathbf{v}_1\mathbf{v}_2\cdots\mathbf{v}_n]$ where $\mathbf{v}_i$ with $i = 1, 2, \ldots, n$ are independent eigenvectors of $\mathbf{A}$ and forming the columns of $\mathbf{W}_R$. We know that each column of $\mathbf{W}_R$ can be chosen fully real-valued, since $\mathbf{A}$ and all eigenvalues are also real (using Lemma 3.3). In this case, the elements of $\mathbf{W}_R^T\mathbf{W}_R$ are:

$$\mathbf{v}_i^T\mathbf{v}_j = \begin{cases} c_i, & i = j; \\ 0, & i \neq j, \end{cases} \tag{3.32}$$

where $c_i > 0 \ \forall i$ and where we have used Lemma 3.3 to obtain the orthogonality of $\mathbf{v}_i$ and $\mathbf{v}_j$ if $i \neq j$. In the case of $i = j$, we have $||\mathbf{v}_i||^2 = c_i$ (with corresponding eigenvalue λ_i), which can also be written as

$$\left|\left|\frac{\mathbf{v}_i}{\sqrt{c_i}}\right|\right|^2 = 1. \tag{3.33}$$

Note that $\frac{\mathbf{v}_i}{\sqrt{c_i}}$ is again an eigenvector of the corresponding eigenvalue λ_i. In other words: each $\mathbf{v}_i$ can be made such that this vector has *unit* length. As a consequence, $\mathbf{W}_R$ is an orthogonal matrix in this case.

□

Theorem 3.2 can even be generalized for the case that $\mathbf{A}$ does *not* have distinct eigenvalues, which is a consequence of Lemma 3.4. In this lemma we use the term 'orthogonally diagonalizable': $\mathbf{P}$ is an *orthogonal matrix* if $\mathbf{P}^{-1} = \mathbf{P}^T$ holds and furthermore, a square matrix $\mathbf{A}$ is *orthogonally diagonalizable*, if there exists an orthogonal matrix $\mathbf{P}$ such that $\mathbf{P}^{-1}\mathbf{A}\mathbf{P}$ is a diagonal matrix.

Lemma 3.4 *An $n \times n$ matrix $\boldsymbol{A}$ is orthogonally diagonalizable if and only if $\boldsymbol{A}$ is a symmetric matrix.*

The proof is difficult and is omitted. The interested reader is referred to O'Nan & Enderton [35] (pp. 405–410). The 'only if'–statement of Lemma 3.4 has also been proved in Golub & Van Loan [23] (Theorem 8.1.1).

3.3 Left and Right Eigenvectors for Complex Matrices

In this section, we assume matrix $\mathbf{A}$ to be complex-valued. This is of our interest, since matrix $\mathbf{A}$ in the linear system $\mathbf{Ap} = \mathbf{f}$ of the DHP is also complex, due to absorbing boundary conditions, see Section 2.5.

3.3.1 Right Eigenvectors

The same theory, as for real-valued $\mathbf{A}$, can be used in the complex case. Hence, we can again construct

$$\mathbf{AW}_R = \mathbf{W}_R \Lambda_R. \tag{3.34}$$

3.3.2 Left Eigenvectors

In the complex case, a left eigenvector for $\mathbf{A}$ satisfies

$$\mathbf{v}_L^H \mathbf{A} = \lambda_L \mathbf{v}_L^H, \tag{3.35}$$

where $\mathbf{v}_L^H$ is called the *Hermitian* of $\mathbf{v}_L$ and is defined by $\mathbf{x}^H \equiv \overline{\mathbf{x}}^T$ for arbitrary vector or matrix $\mathbf{x}$. In other words, $\mathbf{x}^H$ is equal to the transpose of the conjugate of $\mathbf{x}$.

Similar to the right eigenvectors, if $(\lambda_L)_1, (\lambda_L)_2, \ldots, (\lambda_L)_r$ are all eigenvalues and $(\mathbf{v}_L)_1, (\mathbf{v}_L)_2, \ldots, (\mathbf{v}_L)_r$ are the corresponding left eigenvectors of $\mathbf{A}$, then again the set of left eigenvectors forms a basis of a vector space. If this vector space is of dimension n, then we can construct an $n \times n$ matrix $\mathbf{W}_L^H$ whose columns are the components of the left eigenvectors, which has the property that

$$\mathbf{W}_L^H \mathbf{A} = \Lambda_L \mathbf{W}_L^H, \tag{3.36}$$

where Λ_L is again the diagonal matrix with $(\lambda_L)_1, (\lambda_L)_2, \ldots, (\lambda_L)_n$.

3.3.3 Relation Left and Right Eigenvectors

In the same way as in the real-valued case, we can easily show that the left and right eigenvectors can be chosen such that $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$, see Theorem 3.3.

Theorem 3.3 *Let $\boldsymbol{A}$ be a complex diagonalizable matrix. Then $\boldsymbol{W}_L^H$ and $\boldsymbol{W}_R$ can be constructed such that $\boldsymbol{W}_L^H \cdot \boldsymbol{W}_R = \boldsymbol{I}$.*

Proof. We choose $\mathbf{W}_L^H = \mathbf{W}_R^{-1}$, which is possible since $\mathbf{A}$ is diagonalizable. The further proof is analogous to the proof of Theorem 3.1, by using $\mathbf{A}^H$ instead of $\mathbf{A}^T$.

□

Observe again that if both $\mathbf{W}_R$ and $\mathbf{W}_L$ exist, then $\mathbf{W}_L^H \cdot \mathbf{W}_R = \mathbf{I}$ can always be formed, *independent* of the structure of $\mathbf{A}$. The simplest choice is $\mathbf{W}_L^H = \mathbf{W}_R^{-1}$.

If $\mathbf{A}$ has special properties, then $\mathbf{W}_L^H$ can be easier computed. In the next subsections, we deal with the situation when $\mathbf{A}$ is Hermitian and when $\mathbf{A}$ is complex-symmetric.

3.3.4 Properties of Hermitian Matrix

The properties of symmetric matrices in subsection 3.2.4 can be generalized to Hermitian matrices.

Lemma 3.5 *Let $\boldsymbol{A}$ be a Hermitian matrix. Then*

(a) the eigenvalues of $\boldsymbol{A}$ are real;

(b) the eigenvectors corresponding to distinct eigenvalues are orthogonal.

Proof. The proofs are almost identical to the proofs of Lemma 3.3. In these proofs, $\mathbf{A}^H$ has to be used instead of $\mathbf{A}^T$.

□

The following theorem shows how the left eigenvectors can be computed easily, when the right eigenvectors are known.

Theorem 3.4 *Let $\boldsymbol{A}$ be a Hermitian matrix, i.e., $\boldsymbol{A}^H = \boldsymbol{A}$.*

(a) Assume $\boldsymbol{W}_R$ to be the right eigenvector matrix. Then, the left eigenvector matrix can always be chosen such that $\boldsymbol{W}_L = \boldsymbol{W}_R$.

(b) Let $\boldsymbol{A}$ have distinct eigenvalues. Then, there exists a right eigenvector matrix $\boldsymbol{W}_R$ and a left eigenvector matrix $\boldsymbol{W}_L = \boldsymbol{W}_R$ such that these satisfy $\boldsymbol{W}_L^H \boldsymbol{W}_R = \boldsymbol{I}$.

Proof of (a). Since $\mathbf{A}^H = \mathbf{A}$ holds, the following expressions are equivalent:

$$\begin{aligned} \mathbf{A}\mathbf{W}_R &= \mathbf{W}_R\Lambda_R; \\ (\mathbf{A}\mathbf{W}_R)^H &= (\mathbf{W}_R\Lambda_R)^H; \\ \mathbf{W}_R^H\mathbf{A}^H &= \Lambda_R^H\mathbf{W}_R^H; \\ \mathbf{W}_R^H\mathbf{A} &= \Lambda_R\mathbf{W}_R^H; \end{aligned} \tag{3.37}$$

where we have used the fact that diagonal matrix Λ_R consists of only real values (Lemma 3.5). We have derived that $\mathbf{W}_L^H = \mathbf{W}_R^H$, implying $\mathbf{W}_L = \mathbf{W}_R$.

□

Proof of (b). The proof is analogous to the proof of Theorem 3.2.

□

Again, it is possible to generalize this theorem to the case, when the Hermitian matrix does not have distinct eigenvalues, which follows from Lemma 3.6.

Note first that $\mathbf{P}$ is an *unitary matrix* if $\mathbf{P}^{-1} = \mathbf{P}^H$ (thus also $\mathbf{P}^H\mathbf{P} = \mathbf{I}$) and furthermore, a square matrix $\mathbf{A}$ is *unitary diagonalizable* if there exists an unitary matrix $\mathbf{P}$ such that $\mathbf{P}^{-1}\mathbf{A}\mathbf{P}$ is a diagonal matrix.

Lemma 3.6 *Let $\boldsymbol{A}$ be a Hermitian matrix, then $\boldsymbol{A}$ is unitarily diagonalizable.*

The proof is almost the same as the proof of Lemma 3.4, see also O'Nan & Enderton [35].

3.3.5 Properties of Complex-Symmetric Matrix

Let $\mathbf{A}$ be a *complex-symmetric* matrix, i.e.,

$$\mathbf{A}^T = \mathbf{A}, \;\mathbf{A}^H \neq \mathbf{A}. \tag{3.38}$$

In general, this is not an useful symmetry. However, it can be shown that, in this case, we are able to choose $\mathbf{W}_L$ which satisfies $\mathbf{W}_L = \overline{\mathbf{W}}_R$.

Theorem 3.5 *Let $\boldsymbol{A}$ be a complex-symmetric matrix and let $\boldsymbol{W}_R$ be a right eigenvector matrix corresponding to $\boldsymbol{A}$. The left eigenvector matrix $\boldsymbol{W}_L$ can be chosen such that $\boldsymbol{W}_L = \overline{\boldsymbol{W}}_R$.*

Proof. Since $\mathbf{A}^T = \mathbf{A}$ holds, the following expressions are equivalent:

$$\begin{aligned} \mathbf{A}\mathbf{W}_R &= \mathbf{W}_R\Lambda_R; \\ (\mathbf{A}\mathbf{W}_R)^T &= (\mathbf{W}_R\Lambda_R)^T; \\ \mathbf{W}_R^T\mathbf{A}^T &= \Lambda_R^T\mathbf{W}_R^T; \\ \mathbf{W}_R^T\mathbf{A} &= \Lambda_R\mathbf{W}_R^T, \end{aligned} \tag{3.39}$$

leading to $\mathbf{W}_L^H = \mathbf{W}_R^T$ and therefore $\mathbf{W}_L = \overline{\mathbf{W}}_R$.

□

Next, by definition, $\mathbf{A}$ is normal if $\mathbf{A}\mathbf{A}^H = \mathbf{A}^H\mathbf{A}$. Lemma 3.7 can now be proved.

Lemma 3.7 *A square matrix $\boldsymbol{A}$ is unitarily diagonalizable if and only if $\boldsymbol{A}$ is normal.*

The proof (using the Schur decomposition) is omitted here. The reader is referred to Theorem 7.1.3 & Corollary 7.1.4 of Golub & Van Loan [23].

Note that the complex-symmetric matrix $\mathbf{A}$ is *not* unitarily diagonalizable in general, since we usually have $\mathbf{A}\mathbf{A}^H \neq \mathbf{A}^H\mathbf{A}$ and hence, Lemma 3.7 does not hold. However, the eigenvectors of $\mathbf{A}$ are *orthogonal* to each other, assuming that $\mathbf{A}$ has distinct eigenvalues, see the next lemma.

Lemma 3.8 *The eigenvectors corresponding to distinct eigenvalues of a complex-symmetric matrix are orthogonal.*

Proof. The proof is exactly identical to the proof of Lemma 3.3.

Theorem 3.6 *Let $\boldsymbol{A}$ be a complex-symmetric matrix with distinct eigenvalues. Then there exists a $\boldsymbol{W}_R$ and a $\boldsymbol{W}_L = \overline{\boldsymbol{W}}_R$ such that these satisfy $\boldsymbol{W}_L^H\boldsymbol{W}_R = \boldsymbol{I}$.*

Proof. We have to prove that $\mathbf{W}_R^T\mathbf{W}_R = \mathbf{I}$. This can be done almost identical to the proof of Theorem 3.2. Instead of $c_i > 0\ \forall i$ we use $c_i \in \mathbb{C}\backslash\{0\}\ \forall i$.

□

In contrast to the real-symmetric and Hermitian case, it is not clear whether or not Theorem 3.6 can be generalized to the case of matrix $\mathbf{A}$ with non-distinct eigenvalues. This is left for further research.

3.4 Summary of Left and Right Eigenvectors

In this section we summarize the results of the previous two sections about $\mathbf{W}_L$ and $\mathbf{W}_R$.

Case I Assume $\mathbf{A}$ to be a *real* and *diagonalizable* matrix.

- The decomposition $\mathbf{W}_L^T \mathbf{W}_R = \mathbf{I}$ can always be made by choosing $\mathbf{W}_L^T = \mathbf{W}_R^{-1}$.
- In the case of a *real-symmetric* $\mathbf{A}$, we can *even* choose $\mathbf{W}_L = \mathbf{W}_R$ such that $\mathbf{W}_L^T \mathbf{W}_R = \mathbf{I}$ still holds.

Case II Assume that $\mathbf{A}$ to be *complex* and *diagonalizable.*

- The decomposition $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$ can always be made by choosing $\mathbf{W}_L^H = \mathbf{W}_R^{-1}$.
- In the case of a *Hermitian* $\mathbf{A}$, we can *even* choose $\mathbf{W}_L = \mathbf{W}_R$ such that $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$ still holds.
- In the case of a *complex-symmetric*, $\mathbf{A}$ with *distinct* eigenvalues we can choose $\mathbf{W}_L = \overline{\mathbf{W}}_R$ such that $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$ still holds.

In the DHP, we have seen that matrix $\mathbf{A}$ is complex-symmetric (Section 2.5), so the properties given in Section 3.3.5 can be applied. If we use Dirichlet instead of absorbing conditions, matrix $\mathbf{A}$ turns out to be symmetric. In this case the properties of Section 3.2.4 are valid.

Remark

We end with a final remark that sometimes in applications one applies Gram-Schmidt (or other orthogonalization procedures) to modify $\mathbf{W}_R$ or $\mathbf{W}_L$. There are three reasons to do this:

1. scaling $\mathbf{W}_R$ or $\mathbf{W}_L$ such that $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$;
2. forcing $\mathbf{v}_i \perp \mathbf{v}_j$ if $\lambda_i = \lambda_j$;
3. ensuring that $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$ holds, when an approximation of $\mathbf{W}_R$ or $\mathbf{W}_L$ is used.

3.5 Conjugate Inner Product

The inner product $\langle \cdot, \cdot \rangle$ is defined in the following sense.

Definition 3.1 *Let $\boldsymbol{u}, \boldsymbol{v}$ and $\boldsymbol{w}$ be vectors in $\mathbb{C}^n$ and let c be a scalar in $\mathbb{C}$. Then $\langle \cdot, \cdot \rangle$ is an inner product if it satisfies the following properties:*

i. $\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \langle \boldsymbol{v}, \boldsymbol{u} \rangle$;

ii. $\langle \boldsymbol{u} + \boldsymbol{v}, \boldsymbol{w} \rangle = \langle \boldsymbol{u}, \boldsymbol{w} \rangle + \langle \boldsymbol{v}, \boldsymbol{w} \rangle$;

iii. $\langle c\boldsymbol{u}, \boldsymbol{v} \rangle = c \langle \boldsymbol{u}, \boldsymbol{v} \rangle$;

iv. $\langle \boldsymbol{u}, \boldsymbol{u} \rangle \geq 0$ *and* $\langle \boldsymbol{u}, \boldsymbol{u} \rangle = 0$ *if and only if* $\boldsymbol{u} = \boldsymbol{0}$.

The standard complex inner product $\langle \cdot, \cdot \rangle$ is defined by

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x_i \cdot \overline{y}_i, \tag{3.40}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$ and $\mathbf{y} = (y_1, y_2, \ldots, y_n)^T$ are complex-valued vectors. It can be shown that, therefore, this inner product satisfies all conditions of Definition 3.1. However, in this thesis we use the '*conjugate inner product*':

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\text{conj}} = \sum_{i=1}^{n} x_i \cdot y_i. \tag{3.41}$$

Observe that (3.41) is *not* an inner product, but a so-called *semi* inner product since, for some complex vector $\mathbf{x}$, it does not satisfy $\langle \mathbf{x}, \mathbf{x} \rangle \geq 0$ (fourth condition of Definition 3.1). For example, taking $\mathbf{x} = (i, i)^T$, we obtain $\langle \mathbf{x}, \mathbf{x} \rangle = -2$.

In this thesis we write $\langle \cdot, \cdot \rangle$ instead of $\langle \cdot, \cdot \rangle_{\text{conj}}$ for simplicity and we call this the 'conjugate inner product'. Furthermore, in some iterative methods, where we consider the 'real' complex inner products as in (3.40), we denote these with $(\cdot, \cdot)$.

Moreover, in the following if we talk about 'orthogonality' then it means conjugate orthogonality on this conjugate inner product, i.e.,

$$\langle \cdot, \cdot \rangle = \langle \cdot, \cdot \rangle_{\text{conj}} = 0. \tag{3.42}$$

3.6 Solution as Linear Combination of Eigenvectors

In iterative methods, the analysis of eigenvectors often gives a lot of information about the solution. A main theorem, which forms the basis of this analysis, is given in Theorem 3.7.

Theorem 3.7 *Let $\boldsymbol{A}\boldsymbol{p} = \boldsymbol{f}$ be a linear system where $\boldsymbol{f}$ is a given vector of length n and $\boldsymbol{A}$ a non-singular complex-symmetric $n \times n$ matrix with distinct eigenvalues. Then, $\boldsymbol{p}$ can be written as the linear combination of the eigenvectors $\boldsymbol{v}_i$, i.e.,*

$$\boldsymbol{p} = c_1 \boldsymbol{v}_1 + \ldots + c_n \boldsymbol{v}_n, \tag{3.43}$$

where the coefficients c_i are equal to

$$c_i = \frac{\langle \boldsymbol{p}, \boldsymbol{v}_i \rangle}{\langle \boldsymbol{v}_i, \boldsymbol{v}_i \rangle}, \tag{3.44}$$

assuming that $\langle \boldsymbol{v}_i, \boldsymbol{v}_i \rangle \neq 0 \ \forall i = 1, \ldots, n$.

Proof. Using Lemma 3.3, one finds that the eigenvectors corresponding to distinct eigenvalues are orthogonal. Since there are in this case exactly n eigenvectors $\mathbf{v}_i$ which are linear independent by definition, they span the vectorspace $\mathbb{C}^n$. The solution $\mathbf{p}$ is also an element of $\mathbb{C}^n$. Therefore, we can construct

$$\mathbf{p} = c_1\mathbf{v}_1 + \ldots + c_n\mathbf{v}_n, \tag{3.45}$$

with $c_i \in \mathbb{C}$ for all $i = 1, \ldots, n$.

Moreover, for a fixed $i = 1, \ldots, n$, we take the inner product of each side of (3.45) with $\mathbf{v}_i$:

$$\begin{aligned} \langle \mathbf{p}, \mathbf{v}_i \rangle &= \langle c_1\mathbf{v}_1 + \ldots + c_n\mathbf{v}_n, \mathbf{v}_i \rangle \\ &= c_1 \langle \mathbf{v}_1, \mathbf{v}_i \rangle + \ldots + c_n \langle \mathbf{v}_n, \mathbf{v}_i \rangle \\ &= c_i \langle \mathbf{v}_i, \mathbf{v}_i \rangle, \end{aligned} \tag{3.46}$$

since $\langle \mathbf{v}_i, \mathbf{v}_j \rangle = 0$ for $i \neq j$ by orthogonality of the eigenvectors. Hence,

$$c_i = \frac{\langle \mathbf{p}, \mathbf{v}_i \rangle}{\langle \mathbf{v}_i, \mathbf{v}_i \rangle}, \tag{3.47}$$

as claimed.

□

Note that $\langle \mathbf{v}_i, \mathbf{v}_i \rangle = ||\mathbf{v}_i||^2 \neq 0$ does not always hold: let for instance $\mathbf{v} = (1 + i, 1 - i)^T$, then $||\mathbf{v}_i||^2 = 0$.

The consequence of this theorem is that if $k < n$, then $\mathbf{p}_k$ defined by

$$\mathbf{p}_k = c_1\mathbf{v}_1 + \ldots + c_k\mathbf{v}_k, \tag{3.48}$$

is an *approximation* of $\mathbf{p}$. Note that in the case of $k = n$, then it yields $\mathbf{p} = \mathbf{p}_n$.

3.7 Eigenvalue Properties of Special Systems

The theorems, given in this section, will be very useful in our further analysis of eigenvalues in preconditioned systems in Chapters 5 and 8.

We start with Theorem 3.8, which says that the spectrum of **AB** and **BA** are exactly the same.

Theorem 3.8 *Let* $\boldsymbol{A}$ *and* $\boldsymbol{B}$ *be two arbitrary invertible matrices. Then, the systems* $\boldsymbol{AB}$ *and* $\boldsymbol{BA}$ *have the same eigenvalue distribution.*

Proof. Let λ be an arbitrary eigenvalue of **A** with corresponding eigenvector **v**. Then, the following expressions are equivalent:

$$\begin{aligned} \mathbf{ABv} &= \lambda\mathbf{v}; \\ \mathbf{BABv} &= \lambda\mathbf{Bv}; \\ \mathbf{BAw} &= \lambda\mathbf{w}, \end{aligned} \tag{3.49}$$

where $\mathbf{w} = \mathbf{Bv}$ is an eigenvector of **BA** corresponding to λ. Considering (3.49), λ is indeed an eigenvalue of **AB** if and only if λ is also an eigenvalue of **BA**.

□

Note that the definition of 'invertible matrix' forces **A** and **B** in Theorem 3.8 to be *square.*

In Lemma 3.3, it has been proved that a *symmetric* matrix has *real* eigenvalues. We note that if **A** and **B** are symmetric, then in general **AB** and **BA** are *not* symmetric [2]. Therefore, **AB** and **BA** can also have *complex* eigenvalues. However, if we also assume **A** to be positive semi definite (or briefly: PSD), then we can prove that all eigenvalues of **AB** and **BA** are *real*, see Theorem 3.9. In preparation of this theorem, we need the definition and some properties of $\mathbf{A}^{\frac{1}{2}}$.

By definition, an $n \times n$ matrix **A** with real entries is PSD if $\mathbf{x}^T\mathbf{Ax} \geq 0$, for all nonzero vectors **x** with also real entries.

Assume that matrix Λ consists of the eigenvalues of **A** and $\mathbf{W}_R$ is the corresponding right eigenvector matrix such that $\mathbf{A} = \mathbf{W}_R\Lambda\mathbf{W}_R^{-1}$, i.e., **A** is diagonalizable with matrices $\mathbf{W}_R$ and Λ. Then $\mathbf{A}^{\frac{1}{2}}$ is defined by

$$\mathbf{A}^{\frac{1}{2}} = \mathbf{W}_R\Lambda^{\frac{1}{2}}\mathbf{W}_R^{-1}, \tag{3.52}$$

if all diagonal elements of Λ are *non-negative.*

Lemma 3.9 *Let* $\boldsymbol{A}$ *be an invertible matrix which is PSD. Then*

[2] For example, assume we have symmetric matrices:

$$\mathbf{A} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}, \; \mathbf{B} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}. \tag{3.50}$$

Then, matrix **AB** turns out to be non-symmetric:

$$\mathbf{AB} = \begin{bmatrix} 2 & -2 \\ -1 & 4 \end{bmatrix}. \tag{3.51}$$

(a) $\boldsymbol{A}^{\frac{1}{2}}$ *and* $\boldsymbol{A}^{-\frac{1}{2}}$ *exist;*

(b) $\boldsymbol{A}^{\frac{1}{2}}$ *is symmetric, if* $\boldsymbol{A}$ *is also symmetric.*

Proof of (a). Observe that $\mathbf{A}$ is even *positive definite* (PD), since $\mathbf{A}$ is invertible (Theorem 5 of Lay [30]). In this case, all eigenvalues are positive (see Corollary 4.2.3 of Golub & Van Loan). Therefore, matrix Λ consists of positive elements and, by definition, $\mathbf{A}^{\frac{1}{2}}$ and $\mathbf{A}^{-\frac{1}{2}}$ exist.

□

Proof of (b). Matrix $\mathbf{A}^{\frac{1}{2}}$ is symmetric, since

$$\begin{aligned}\left(\mathbf{A}^{\frac{1}{2}}\right)^T &= \left(\mathbf{W}_R\Lambda^{\frac{1}{2}}\mathbf{W}_R^{-1}\right)^T = \mathbf{W}_R^{-T}\Lambda^{\frac{1}{2}T}\mathbf{W}_R^T \\ &= \left(\mathbf{W}_R^T\right)^T\Lambda^{\frac{1}{2}}\mathbf{W}_R^{-1} = \mathbf{W}_R\Lambda^{\frac{1}{2}}\mathbf{W}_R^{-1} = \mathbf{A}^{\frac{1}{2}},\end{aligned} \tag{3.53}$$

using the fact that $\mathbf{W}_R^T\mathbf{W}_R = \mathbf{I}$ (Lemma 3.4).

□

Now, the following theorem can be proved.

Theorem 3.9 *Let* $\boldsymbol{A}$ *and* $\boldsymbol{B}$ *be invertible matrices. Moreover, let* $\boldsymbol{A}$ *be also PSD. Then*

(a) $\boldsymbol{BA}$ *has eigenvalues which are all real;*

(b) $\boldsymbol{AB}$ *has also only real-valued eigenvalues.*

Proof of (a). Since $\mathbf{A}$ is PSD, matrix $\mathbf{A}^{\frac{1}{2}}$ exists and is also symmetric (Lemma 3.9). Now, each arbitrary eigenvalue λ with corresponding eigenvector $\mathbf{v}$ of $\mathbf{BA}$ satisfies

$$\mathbf{BAv} = \lambda\mathbf{v}. \tag{3.54}$$

The above expression is equivalent with

$$\begin{aligned}\mathbf{A}^{\frac{1}{2}}\mathbf{BAv} &= \lambda\mathbf{A}^{\frac{1}{2}}\mathbf{v}; \\ \mathbf{A}^{\frac{1}{2}}\mathbf{BAA}^{-\frac{1}{2}}\mathbf{A}^{\frac{1}{2}}\mathbf{v} &= \lambda\mathbf{A}^{\frac{1}{2}}\mathbf{v}; \\ \mathbf{A}^{\frac{1}{2}}\mathbf{BAA}^{-\frac{1}{2}}\mathbf{w} &= \lambda\mathbf{w}; \\ \mathbf{A}^{\frac{1}{2}}\mathbf{BA}^{\frac{1}{2}}\mathbf{w} &= \lambda\mathbf{w},\end{aligned} \tag{3.55}$$

where $\mathbf{w} = \mathbf{A}^{\frac{1}{2}}\mathbf{v}$ is an eigenvector of $\mathbf{A}^{\frac{1}{2}}\mathbf{BA}^{\frac{1}{2}}$. Observe now that $\mathbf{A}^{\frac{1}{2}}\mathbf{BA}^{\frac{1}{2}}$ is symmetric, since

$$\left(\mathbf{A}^{\frac{1}{2}}\mathbf{BA}^{\frac{1}{2}}\right)^T = \mathbf{A}^{\frac{1}{2}T}\mathbf{B}^T\mathbf{A}^{\frac{1}{2}T} = \mathbf{A}^{\frac{1}{2}}\mathbf{BA}^{\frac{1}{2}}. \tag{3.56}$$

The consequence is that each eigenvalue λ is real-valued (Lemma 3.3). Therefore, matrix $\mathbf{BA}$ has indeed real eigenvalues.

□

Proof of (b). The proof is almost identical to the proof of (a).

Observe first that, since $\mathbf{A}$ is PSD, matrix $\mathbf{A}^{-\frac{1}{2}}$ exists and is also symmetric (Lemma 3.9). Next, each eigenvalue λ with corresponding eigenvector $\mathbf{v}$ of the system $\mathbf{AB}$ has to satisfy

$$\mathbf{ABv} = \lambda \mathbf{v}. \tag{3.57}$$

Multiplying both sides with $\mathbf{A}^{-\frac{1}{2}}$ leads to

$$\mathbf{A}^{\frac{1}{2}}\mathbf{Bv} = \lambda \mathbf{A}^{-\frac{1}{2}}\mathbf{v}, \tag{3.58}$$

which can be rewritten in

$$\mathbf{A}^{\frac{1}{2}}\mathbf{B}\mathbf{A}^{\frac{1}{2}}\mathbf{w} = \lambda \mathbf{w}, \tag{3.59}$$

where $\mathbf{w} = \mathbf{A}^{-\frac{1}{2}}\mathbf{v}$. Now, the matrix $\mathbf{A}^{\frac{1}{2}}\mathbf{B}\mathbf{A}^{\frac{1}{2}}$ is symmetric (see Eq. (3.56)) and therefore all eigenvalues λ have to be *real*-valued.

□

The last lemma of this chapter deals with eigenvalues of $\mathbf{A}$ and its conjugate $\overline{\mathbf{A}}$.

Lemma 3.10 *Let $\boldsymbol{A}$ be an complex diagonizable matrix. If λ is an eigenvalue of $\boldsymbol{A}$, then its conjugate $\overline{\lambda}$ is an eigenvalue of $\overline{\boldsymbol{A}}$.*

Proof. Let λ being an eigenvalue and $\mathbf{v}$ an corresponding eigenvector of $\mathbf{A}$. Then the following statements are equivalent:

$$\begin{array}{rcl} \mathbf{Av} & = & \lambda \mathbf{v}; \\ \overline{\mathbf{Av}} & = & \overline{\lambda \mathbf{v}}; \\ \overline{\mathbf{A}}\mathbf{w} & = & \overline{\lambda}\mathbf{w}, \end{array} \tag{3.60}$$

where $\mathbf{w} = \overline{\mathbf{v}}$ is the corresponding eigenvector of $\overline{\lambda}$.

□

Chapter 4

Iterative Methods and Preconditioners

In Chapter 2, we have seen that the 2-dimensional (discrete) Helmholtz problem (HP) leads to the following linear system:

$$\mathbf{Ap} = \mathbf{f}, \quad \mathbf{A} \in \mathbb{C}^{n \times n}, \tag{4.1}$$

where $\mathbf{A}$ is an $n \times n$ matrix and $\mathbf{p}$ and $\mathbf{f}$ are both complex vectors with n elements. Moreover, matrix $\mathbf{A}$ is sparse and complex-symmetric. We intend to solve (4.1) where $\mathbf{p}$ is the unknown vector.

In general, this linear system (4.1) can be solved by *direct* numerical methods, like *Gauss-elimination* or *nested dissection method.* A case with 1000 × 1000 gridpoints has been tested on a workstation and the nested dissection method can indeed be applied to solve the linear system efficiently, see Plessix & Mulder [37]. In 3-dimensional cases, direct methods are in general *inefficient* to use. For instance, the nested dissection method gives problems because the amount of fill-in is too large, see Erlangga [14].

However, solution methods with an acceptable efficiency can still be pursued by implementing *iterative* numerical methods for the linear system (4.1). Recently, several iterative methods have been developed. These methods are based on *Krylov subspaces*, see e.g. Saad [39]. In Section 4.1 of this chapter, we first describe these Krylov subspaces shortly.

The *conjugate gradient* (CG) method is a so-called 'Krylov' method which is the most popular one. Moreover, the algorithm of the CG method is of importance as a basis for deriving several related Krylov iterative methods. For a full treatment of CG, one is referred to Saad [39], Vuik [53] or Shewchuk [41]. However, the CG method is not applicable to solve the HP, because the linear system has to be real and positive-definite to obtain an accurate solution, see e.g. section 6.7 of [39].

In Section 4.2 of this chapter, we describe some Krylov methods relevant to the HP. Since we aim at the numerical solution of a *complex-symmetric* and (strongly) *indefinite* linear system, we consider only iterative methods feasible for this kind of linear systems. Bi-CGSTAB, GMRES and GCR are the Krylov iterative methods, which are treated in this chapter.

In practice, standard iterative methods are not sufficiently efficient for solving a sparse and large linear system without modifications. It is known (see e.g. [39, 53]) that in order to obtain a very efficient algorithm, the linear system should be transformed into a formulation which is identical and therefore gives the same solution, but which is much faster to solve. This process is called *preconditioning.* Without this process, Krylov iterative methods are inattractive. Therefore, we treat some preconditioners in Section 4.3.

4.1 Krylov Subspaces

In standard iterative methods, the iterant in the $(j+1)$-th step can be determined from the j-th step in the following way:

$$\mathbf{p}_{j+1} = \mathbf{p}_j + \mathbf{M}^{-1}\mathbf{r}_j, \tag{4.2}$$

where the preconditioner $\mathbf{M}$ is an $n \times n$ matrix and the j-th *residual* $\mathbf{r}_j$ is defined as

$$\mathbf{r}_j = \mathbf{f} - \mathbf{A}\mathbf{p}_j, \tag{4.3}$$

which is a measure of the difference of the iterative and the exact solution of the problem. If we work out the first iterations of (4.2), one obtains

$$\left\{\begin{array}{lcl} \mathbf{p}_0 & & \\ \mathbf{p}_1 & = & \mathbf{p}_0 + \mathbf{M}^{-1}\mathbf{r}_0 \\ \mathbf{p}_2 & = & \mathbf{p}_1 + \mathbf{M}^{-1}(\mathbf{f} - \mathbf{A}\mathbf{p}_0 - \mathbf{A}\mathbf{M}^{-1}\mathbf{r}_0) \\ & = & \mathbf{p}_0 + 2\mathbf{M}^{-1}\mathbf{r}_0 - \mathbf{M}^{-1}\mathbf{A}\mathbf{M}^{-1}\mathbf{r}_0 \\ \mathbf{p}_3 & = & \cdots \\ & \vdots & \end{array}\right. \tag{4.4}$$

Using (4.4), we get the following expression for (4.2):

$$\mathbf{p}_{j+1} \in \mathbf{p}_0 + \text{ span}\left\{\mathbf{M}^{-1}\mathbf{r}_j, \mathbf{M}^{-1}\mathbf{A}(\mathbf{M}^{-1}\mathbf{r}_j), \ldots, (\mathbf{M}^{-1}\mathbf{A})^{j-1}(\mathbf{M}^{-1}\mathbf{r}_j)\right\}. \tag{4.5}$$

Subspaces of the form

$$\mathcal{K}_j(\mathbf{A}, \mathbf{r}_0) = \text{ span}\left\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \mathbf{A}^2\mathbf{r}_0, \ldots, \mathbf{A}^{j-1}\mathbf{r}_0\right\}, \tag{4.6}$$

are called *Krylov subspaces* with dimension j, belonging to $\mathbf{A}$ and $\mathbf{r}_0$. Using (4.6), we get the following expression for standard iterative methods:

$$\mathbf{p}_{j+1} \in \mathbf{p}_0 + \mathcal{K}_j(\mathbf{M}^{-1}\mathbf{A}, \mathbf{M}^{-1}\mathbf{r}_0). \tag{4.7}$$

Expression (4.7) is in fact equivalent to (4.2) and (4.5).

From (4.7), we can observe that Krylov subspace methods rely on finding a matrix $\mathbf{M}$ and a basis for $\mathcal{K}_j$, such that the iterative method converges fast with reasonable accuracy and efficiency with respect to memory and computational time.

Standard iterative methods like *Gauss-Jacobi* and *Gauss-Seidel* are described in [39, 53]. If we denote $\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{L}^T$ with $\mathbf{D}$ the matrix with the main diagonal of $\mathbf{A}$ and $\mathbf{L}$ the (strict) lower triagonal part of $\mathbf{A}$, then we get the following expressions for matrix $\mathbf{M}$:

- Gauss-Jacobi (GJ): $\mathbf{M}_{GJ} = \mathbf{D}$;
- Gauss-Seidel (GS): $\mathbf{M}_{GS} = \mathbf{D} - \mathbf{L}$.

It is well-known that these methods converge very slowly in (3-dimensional) practical problems. However, the residuals during the GS process decrease fast in the first iterations. Due to this nice property, the GS method is used in our successive refinement methods, see Chapter 11 and Appendix K.

4.2 Krylov Iterative Methods

As earlier mentioned, the CG method is the most prominent Krylov iterative method for solving a sparse system $\mathbf{Ap} = \mathbf{f}$, where $\mathbf{A}$ has to be *Hermitian* and *positive definite*. However, matrix $\mathbf{A}$ in the HP is neither Hermitian nor positive definite, in general.

The biconjugate gradient method (Bi-CG) takes another approach which is applicable for non-Hermitian and indefinite systems, see Fletcher [17]. This method replaces the orthogonal sequence of residuals by two mutually orthogonal sequences, at the price of *no* longer providing a minimization, but with the advantage that it can be used for non-Hermitian systems with short recurrences.

The conjugate gradient squared (CGS) method was developed in 1984 by Sonneveld [47], where a 'contraction' operator in Bi-CG is applied twice. Often, one observes a speed of convergence for CGS that is about twice as fast as for the Bi-CG method, which is in agreement with the observation that the same contraction operator is applied twice. Moreover, CGS requires about the same number of operations per iteration as the Bi-CG method, but it does not involve computations with $\mathbf{A}^H$. Hence, in circumstances where computation with $\mathbf{A}^H$ is impractical, CGS may be attractive.

In this section, we describe shortly other Krylov iterative methods (respectively Bi-CGSTAB, GMRES and GCR) which are used to solve the HP in this thesis. Bi-CGSTAB and GMRES are as subroutines available in MATLAB, while we have to implement GCR in MATLAB before we are able to use it.

4.2.1 Bi-CGSTAB Method

The CGS algorithm is based on squaring the residual polynomial and therefore, it shows often *irregular* convergence patterns which may lead to substantial build-up of rouding errors, see Van der Vorst [49]. The *biconjugate gradient stabilized* (Bi-CGSTAB) algorithm is a variation of CGS, which was developed by Van der Vorst [49] to remedy this difficulty. Instead of computing the residual vector $\tilde{\mathbf{r}}_j = \mathcal{R}_j^2(\mathbf{A})\tilde{\mathbf{r}}_0$ (where $\mathcal{R}_j(\mathbf{A})$is the j-th degree polynomial in $\mathbf{A}$) in CGS, Bi-CGSTAB computes

$$\tilde{\mathbf{r}}_j = \mathcal{Q}_j(\mathbf{A})\mathcal{R}_j(\mathbf{A})\mathbf{r}_0, \tag{4.8}$$

with $\mathcal{Q}_j(\mathbf{A})$ a new polynomial which is defined recursively at each step, with the goal of '*stabilizing*' or '*smoothing*' the convergence behavior of the original algorithm. Specifically, $\mathcal{Q}_j$ is defined by the simple recurrence

$$\mathcal{Q}_{j+1}(t) = (1 - \omega_j t)\mathcal{Q}_j. \tag{4.9}$$

This is equivalent with

$$\mathcal{Q}_{j+1}(t) = (1-\omega_0 t)(1-\omega_1 t)\cdots(1-\omega_j t), \tag{4.10}$$

in which the scalars ω_k for all $k = 0, 1, \ldots, j$ are to be determined. This is done by minimizing $\tilde{\mathbf{r}}_j$ as in (4.8) with respect to the scalars ω_k.

The complete derivation of Bi-CGSTAB can be found in [39, 49]. The sketch of the resulting algorithm is given below.

Algorithm 4.1: Biconjugate Gradient Stabilized (Bi-CGSTAB)

```
Compute $\mathbf{r}_0 := \mathbf{f} - \mathbf{A}\mathbf{p}_0$
Set $\mathbf{z}_0 := \mathbf{r}_0$, $\alpha_0 := \frac{\tilde{\mathbf{z}}_0}{(\mathbf{A}\mathbf{z}_0, \tilde{\mathbf{r}}_0)}$ and $\mathbf{s}_0 := \mathbf{r}_0 - \alpha_0 \mathbf{A}\mathbf{z}_0$
Choose $\tilde{\mathbf{r}}_0$ arbitrary

For $j := 0, 1, \ldots,$ until convergence Do :
    $\mathbf{w}_j := \mathbf{A}\mathbf{z}_j$
    $\mathbf{v}_j := \mathbf{A}\mathbf{s}_j$
    $\alpha_j := \frac{\tilde{\mathbf{z}}_j}{(\mathbf{w}_j, \tilde{\mathbf{r}}_0)}$
    $\mathbf{s}_j := \mathbf{r}_j - \alpha_j \mathbf{w}_j$
    $\omega_j := \frac{(\mathbf{v}_j, \mathbf{s}_j)}{(\mathbf{v}_j, \mathbf{v}_j)}$
    $\mathbf{p}_{j+1} := \mathbf{p}_j + \alpha_j \mathbf{z}_j + \omega_j \mathbf{s}_j$
    $\mathbf{r}_{j+1} := \mathbf{s}_j - \omega_j \mathbf{v}_j$
    $\beta_j := \frac{\alpha_j}{\omega_j} \frac{\rho_{j+1}}{\rho_j}$
    $\mathbf{z}_{j+1} := \mathbf{r}_j + \beta_j(\mathbf{z}_j - \omega_j \mathbf{w}_j)$
EndFor
```

The exact definition of the quantities $\tilde{\mathbf{z}}_j$ and ρ_j are omitted here, but they can be found in Saad [39].

The advantage of the method is that it uses *short* recurrences, but, unfortunately, it is based on a *semi-optimality* property. As a result, more matrix-vector products are needed and no convergence properties have been proved. One observes in the algorithm that Bi-CGSTAB requires two matrix-vector products and four inner products, i.e., two inner products more than the biconjugate gradient method or the conjugate gradient squared method.

Investigation of the Bi-CGSTAB algorithm has been reported in Van der Vorst [49] for various applications and compared to Bi-CG and CGS. In general, Bi-CGSTAB converges *more smoothly* than CGS of Bi-CG. However, the convergence rate is typically the same. In some non-Hermitian cases, it is revealed that when CGS fails to converge and shows spurious irregularity, Bi-CGSTAB still converges. The convergence rate is also faster than CGS and Bi-CG.

Though Bi-CGSTAB is an attractive alternative to CGS, further investigation reveal a weakness of this algorithm, as mentioned in Erlangga [14]. If the parameters ω_k becomes very close to zero during the recursion, the algorithm may stagnate or break down. Numerical experiments confirm that this is likely to happen if $\mathbf{A}$ is real and has complex eigenvalues with imaginary part larger than the real part. To overcome this, improvements to Bi-CGSTAB have been proposed, resulting in Bi-CGSTAB(l) with $l \in \mathbb{N}$, see Sleijpen and Fokkema [43]. This is a modification by forming a general l-order minimum-residual polynomial, instead of using $l = 1$ in the original Bi-CGSTAB.

Finally, after convergence reached with the original Bi-CGSTAB, it is always necessary to compare the norm of the updated residual to the exact residual $||\mathbf{f} - \mathbf{A}\mathbf{p}_k||_2$, see Vuik [53]. If 'near' break down had occurred, then these quantities may be different by several orders of magnitude. In such a case, methods as Bi-CGSTAB(l) should be applied.

4.2.2 GMRES Method

In Bi-CGSTAB, we have seen that it is based on short recurrences and on only a semi-optimality property. Another kind of Krylov methods is the (full) *generalized minimal residual* (GMRES) method (Saad & Schultz [40]), which minimizes the residual norm over the Krylov subspace. In order to reach this, it generates a sequence of orthogonal vectors with *long* recurrences, due to the non-Hermitian matrix $\mathbf{A}$.

The method applies a variant of the *Arnoldi's procedure* (Arnoldi [4]) to find a set of orthonormalized vectors. This procedure and some properties are given in **Appendix A**.

The complete derivation of GMRES can be found in [39, 40]. The sketch of the resulting GMRES-algorithm is given below.

Algorithm 4.2: (Full) Generalized Minimum Residual (GMRES)

```
1. Choose $\mathbf{x}_0$ and compute $\mathbf{r}_0 := \mathbf{f} - \mathbf{A}\mathbf{p}_0$, $\beta := ||\mathbf{r}_0||_2$ and $\mathbf{v}_1 := \mathbf{r}_0/\beta$
2. Define the $(m+1) \times m$ matrix $\mathbf{H}_m := \{h_{i,j}\}_{1 \le i \le m+1, 1 \le j \le m}$. Set $\mathbf{H}_m := 0$

3. For $j := 1, 2, \ldots,$ until convergence Do :
4.       $\mathbf{w}_j := \mathbf{A}\mathbf{v}_j$

5.       For $i := 1, 2, \ldots, j$ Do :
6.             $h_{i,j} := (\mathbf{w}_j, \mathbf{v}_i)$
7.             $\mathbf{w}_j := \mathbf{w}_j - h_{ij}\mathbf{v}_i$
8.       EndFor

9.       $h_{j+1,j} := ||\mathbf{w}_j||_2$
10.      $\mathbf{v}_{j+1} := \frac{\mathbf{w}_j}{h_{j+1,j}}$

11. EndFor
12. Compute $\mathbf{y}_m := \arg\min_{\mathbf{y}} ||\beta e_1 - \bar{\mathbf{H}}_m \mathbf{y}||_2$
13. Compute $\mathbf{p}_m := \mathbf{p}_0 + \mathbf{V}_m \mathbf{y}_m$
```

Line 2 to 10 represent the Arnoldi's algorithm for orthogonalization. Moreover, the quantities $\mathbf{H}_m, \mathbf{V}_m$ and e_1 are defined as in Proposition A.1.

The GMRES algorithm may break down if $h_{j+1,j} = 0$ at iteration step j (see line 9). However, this situation implies that the residual vector is zero and therefore, the algorithm gives the exact solution at this step. Hence, examination of value $h_{j+1,j}$ becomes important.

For GMRES, we see in many cases a *super-linear* convergence behavior comparable to CG. Recently, some of these results are proved for GMRES by Van der Vorst & Vuik [50].

If we denote the dimension of the square matrix $\mathbf{A}$ with n, then the GMRES method (like any orthogonalizing Krylov subspace method) will converge in no more than n steps, where we consider exact arithmetics. In practical situation and thus also in the HP, iteration number n is *large* and therefore the GMRES algorithm becomes *impractical*, as a consequence of a lack of memory and increasing computational requirements. This is understandable from the fact that, during the Arnoldi steps (lines 2-10), the number of vectors requiring storage increases. There are several ways to remedy this problem, like *restarting* and *truncating*, see Saad [39] or Vuik [53].

The restarted GMRES method is denoted by GMRES(m), where one stops the full GMRES after m iterations to form the approximated solution and, thereafter, one applies this as starting vector for a following application of GMRES. However, restarting leads to the break of many of the nice properties of the full GMRES. For instance, the optimality property is only valid inside a GMRES(m) step and the super-linear convergence behavior is lost, which are severe drawbacks of the GMRES(m) method.

4.2.3 GCR Method

Slightly earlier than the GMRES method, Eisenstat, Elman & Schultz [13] proposed the *generalized conjugate residual* (GCR) method. This method generates also a sequence of orthogonal vectors with *long* recurrences and it is based on an *optimal* property.

For the full derivation and other properties like convergence results of the GCR method, we refer to [13]. The GCR-algorithm is given as follows:

Algorithm 4.3: Generalized Conjugate Residual (GCR)

```
1.  Choose $\mathbf{p}_0$ and compute $\mathbf{r}_0 = \mathbf{f} - \mathbf{A}\mathbf{p}_0$

2.  For $j := 1, 2, \ldots,$ until convergence Do :
3.      $\mathbf{s}_j := \mathbf{r}_{j-1}$
4.      $\mathbf{v}_j := \mathbf{A}\mathbf{s}_j$

5.      For $i := 1, 2, \ldots, j-1$ Do :
6.          $\alpha := (\mathbf{v}_j, \mathbf{v}_i)$
7.          $\mathbf{v}_j := \mathbf{v}_j - \alpha\mathbf{v}_i,\ \mathbf{s}_j := \mathbf{s}_j - \alpha\mathbf{s}_i$
8.      EndFor

9.      $\mathbf{v}_j := \frac{\mathbf{v}_j}{||\mathbf{v}_j||_2},\ \mathbf{s}_j := \frac{\mathbf{s}_j}{||\mathbf{v}_j||_2}$
10.     $\mathbf{p}_j := \mathbf{p}_{j-1} + (\mathbf{r}_{j-1}, \mathbf{v}_j)\,\mathbf{s}_j$
```

```
    $\mathbf{r}_j := \mathbf{r}_{j-1} + (\mathbf{r}_{j-1}, \mathbf{v}_j)\, \mathbf{v}_j$
EndFor
```

The vectors $\mathbf{s}_j$ and $\mathbf{v}_j$ cost two times as much memory as for GMRES, while the rate of convergence of GCR and GMRES are comparable, i.e., the number of iterations using full GCR and GMRES are approximately the same, which is in fact the result of optimizing the same norm and choosing the same search directions. However, there are examples where GCR breaks down, while GMRES still converges.

When the required memory is not available, the GCR method can be restarted or truncated as in GMRES. In general, we see that truncated GCR methods have a better convergence behavior, especially if super-linear convergence plays an important role. Therefore, if restarting or truncation is necessary, truncated GCR is generally better than restarted GMRES (see Vuik [53]).

4.2.4 Starting Vector and Termination Criterium

In Algorithms 4.1–4.3, the starting vector $\mathbf{p}_0$ and the termination criterium are still undefined. In this subsection we specify these quantities.

Starting Vector

We note first that the choice of the startvector $\mathbf{p}_0$ can be *crucial* in results. For instance, if one chooses a starting vector relatively close to the solution, the iterative method converges fast in general. There are several methods available to choose a starting vector suitable, see [39, 53].

We do not pay attention to starting vectors, since in this thesis we are mainly interested in properties of preconditioners which are in general independent of starting vectors. In all testruns of this thesis, if the starting vector is not defined, we start with

$$\mathbf{p}_0 = \mathbf{0}, \tag{4.11}$$

in other words: the zero vector is used as starting vector in all algorithms.

Termination Criterium

In Algorithms 4.1–4.3 of the Krylov iterative methods, no criteria have been given to stop the iterative process. In general, the iterative method should be stopped if the approximate solution is accurate enough. A good termination criterion is very important for an iterative method. If the criterion is too weak, the approximate solution is useless, whereas a too severe criterion gives an iterative solution method which never stops or costs too much work. Several termination criteria can be found in [39, 53].

In this thesis we apply the following termination criterium:

$$\vartheta(\mathbf{p}_i, \epsilon) := \frac{||\mathbf{f} - \mathbf{A}\mathbf{p}_i||_2}{||\mathbf{f}||_2} < \epsilon \tag{4.12}$$

where we call $\vartheta(\mathbf{p}_i, \epsilon)$ the *relative residual* (RR) criterium and moreover, $\epsilon > 0$ is called the *tolerance* of the method. For an accurate solution of the HP, we choose

$$\epsilon = 10^{-6}. \tag{4.13}$$

Now, in each algorithm of 4.1–4.3, we replace the line

$$\textbf{For } j := 1, 2, \ldots, \ \textit{until convergence} \ \textbf{Do} \ , \tag{4.14}$$

by the new line

$$\textbf{For } j := 1, 2, \ldots, \ \textit{until} \ \vartheta(\mathbf{p}_i, \epsilon) \ \textbf{Do} \ . \tag{4.15}$$

4.3 Preconditioners

Lack of *robustness* and *efficiency* are widely recognized weaknesses of iterative solvers, compared to direct solvers. This is mainly the consequence of the fact that the convergence behavior of Krylov subspace methods depends strongly on the eigenvalue distribution of the coefficient matrix $\mathbf{A}$.

Both robustness and efficiency can be improved by using *preconditioning*. Preconditioning is simply a means of transforming the original linear system into one which has the *same* solution, but which is likely to be *faster* to solve with an iterative solver.

If we solve $\mathbf{Ap} = \mathbf{f}$ with a preconditioner $\mathbf{M}$, then we could solve the following preconditioned system:

$$\mathbf{M}^{-1}\mathbf{Ap} = \mathbf{M}^{-1}\mathbf{f}, \tag{4.16}$$

where we assume $\mathbf{M}$ and $\mathbf{A}$ being matrices with sizes $n \times n$. This preconditioned system (4.16) is known as a *left preconditioned system*. Another preconditioned systems [1] can be found in Saad [39].

In **Appendix B** one can find the *preconditioned* Bi-CGSTAB, GMRES and GCR, which are slightly different compared to Algorithms 4.1–4.3.

We are looking for a preconditioner $\mathbf{M}$ such that (4.16) is faster to solve, relative to the original system. The ideal choice is $\mathbf{M} = \mathbf{A}$, which is obviously impractical, since in general the inverse of $\mathbf{A}$ is expensive to compute. A good preconditioner has to satisfy the following requirements:

(i) The system $\mathbf{Mx} = \mathbf{b}$, with $\mathbf{b}$ a known vector and $\mathbf{x}$ an unknown vector of length n, should be solvable at low cost;

(ii) the eigenvalues of $\mathbf{M}^{-1}\mathbf{A}$ should be clustered (around 1).

The second requirement can be explained as follows. A linear system obtained from discretizations of a PDE can have a strongly distributed spectrum and can result in an indefinite system, i.e., the spectrum consists of both positive and negative real eigenvalues. For such problems, the iterative methods show slow convergence or even breakdown. A good preconditioner, which leads to fast convergence, can transform the original linear system into a system with a *clustered* spectrum, i.e., the spectrum consists of eigenvalues which are concentrated in a close region. It is also important that a preconditioned system does *not* have eigenvalues close to zero, which causes also slow convergence generally.

We can distinguish two approaches for constructing preconditioners:

- *Matrix*-based approach. Within this class we have for instance:

[1] In general, we aim to find $\mathbf{M}_L$ and $\mathbf{M}_R$ such that

$$\mathbf{M}_L^{-1}\mathbf{A}\mathbf{M}_R^{-1}\mathbf{y} = \mathbf{M}_L^{-1}\mathbf{f}, \quad \mathbf{p} = \mathbf{M}_R^{-1}\mathbf{y}. \tag{4.17}$$

is easier to solve. In this thesis, the diagonal (D) and incomplete cholesky (IC) preconditioners apply (4.17) to force a symmetric preconditioned system, which can be favorable in iterative methods.

  - (**D**): *diagonal* preconditioner (Van der Sluis [44]),
  - (**IC**): *incomplete Choleski* preconditioners (e.g. Made [31]);
  - (**ILU**(p)): several variants of preconditioners based on *incomplete LU-factorization* (see Saad [39]).

- *Operator*-based approach. Examples of this kind of preconditioners are:
  - (**CSL**): *complex shifted Laplace* preconditioner (Erlangga, Vuik & Oosterlee [15, 52]);
  - (**AILU**): *analytic ILU* preconditioner (Gander & Nataf [18]);
  - (**SoV**): preconditioner based on *separation of variables* (Plessix & Mulder [37]).

In general, the *reliability* of iterative techniques, when dealing with various applications, depends much more on the *quality* of the preconditioner than on the particular Krylov subspace accelerators used.

As earlier mentioned in Chapter 1, we consider mainly the SoV and CSL preconditioners in this thesis. In the next subsections, we give a short description of these operator-based preconditioners. Moreover, in **Appendix C** the D and IC preconditioners are described, which are used in this thesis to compare some results with SoV and CSL in Chapter 6.

4.3.1 CSL Preconditioner

In Chapter 2, we have seen that the linear system $\mathbf{Ap} = \mathbf{f}$ is derived from:

$$\begin{cases} -\Delta p(\mathbf{x}) - k^2(\mathbf{x})p(\mathbf{x}) &= f(\mathbf{x}), \quad \mathbf{x} = (x, y) \in \Omega; \\ \\ \frac{\partial}{\partial n} p(\mathbf{x}) + ikp(\mathbf{x}) &= 0, \qquad \mathbf{x} \in \partial\Omega, \end{cases} \tag{4.18}$$

where we have taken domain Ω as a computational box, i.e., $\Omega = (0, L)^2$, $L > 0$.

Now, the Helmholtz operator in (4.18) can be written as:

$$\mathcal{L}_H = -\Delta - k^2(\mathbf{x}), \tag{4.19}$$

and in fact, we can say that $\mathbf{A}$ is based on $\mathcal{L}_H$. More concretely, matrix $\mathbf{A}$ can be splitted into two parts: the Laplace matrix $-\mathbf{B}$ and the additional diagonal matrix $k^2(\mathbf{x})\mathbf{I}$ such that

$$\mathbf{A} = -\mathbf{B} - k^2(\mathbf{x})\mathbf{I}. \tag{4.20}$$

Next, we introduce a complex coefficient of the form $\alpha + i\beta$ and we define the following so-called *Shifted Laplace* (SL) operator:

$$\mathcal{L}_{SL} = -\Delta + (\alpha + \beta i)k^2(\mathbf{x}), \tag{4.21}$$

where $\alpha, \beta \in \mathbb{R}$ are parameters with $\alpha \geq 0$. Then the SL-preconditioner $\mathbf{M}_{SL}$ is based on $\mathcal{L}_{SL}$ and we can write:

$$\mathbf{M}_{SL} = -\mathbf{B} + (\alpha + \beta i)k^2(\mathbf{x})\mathbf{I}. \tag{4.22}$$

Note that the condition $\alpha \geq 0$ is required to ensure $\mathbf{M}_{SL}$ being positive semi-definite (PSD), such that for instance multigrid can be applied to solve $\mathbf{Mx} = \mathbf{b}$.

Well-known choices for parameters α, β are

- $\alpha = 0$ and $\beta = 0$ (Bayliss, Goldstein & Turkel [6]);
- $\alpha = 1$ and $\beta = 0$ (Laird [29]);

However, Erlangga, Vuik & Oosterlee [15, 52] have shown for the problem with *Dirichlet* boundaries that the choices

$$\alpha = 0 \quad \text{and} \quad \beta = 1, \tag{4.23}$$

leading to the *Complex Shifted Laplace* (CSL) preconditioner $\mathbf{M}_{CSL}$:

$$\mathbf{M}_{CSL} = -\mathbf{B} + ik^2(\mathbf{x})\mathbf{I}, \tag{4.24}$$

shows better results than the earlier mentioned choices of Laird and Bayliss *et.al.* More strongly: The choices $\alpha = 0$ and $\beta = 1$ are the *optimal* choices of the parameters [2], analyzing the eigenvalue distribution of $\mathbf{M}_{CSL}^{-1}\mathbf{A}$ with Dirichlet boundary conditions, see [15, 52].

More details about the CSL preconditioner can be found in [15, 52], including some properties about the spectra and condition numbers.

4.3.2 SoV Preconditioner

Plessix & Mulder [37] have proposed a preconditioner based on separation of variables (SoV).

In (4.19), we have seen that the Helmholtz operator can be viewed as a Laplace operator $-\Delta$ with an additional term $-k^2(\mathbf{x})$. For the Laplace equation (with or without an additional *constant* term), an analytic solution can be obtained using the separation–of–variables method, see for instance Heikkola, Kuznetsov & Lipnikov [25] and Rossi & Toivanen [38]. One may consider the same solution procedure, which could work nicely for the Helmholtz equation. However, the presence of the inhomogeneous wavenumber $k(\mathbf{x})$ actually prevents application of the same method on the latter problem.

Fortunately, we can decompose $k(\mathbf{x})$ into a formulation suitable for a separation–of–variables method, which can be applied as a preconditioner for solving the Helmholtz equation. Wavenumber $k(\mathbf{x})$ is decomposed in the following way:

$$k^2(\mathbf{x}) = k_x^2(x) + k_y^2(y) + \tilde{k}^2(x, y), \tag{4.25}$$

where $\tilde{k}(x, y)$ satisfies the conditions:

$$\begin{cases} \int_x \tilde{k}^2(x, y)\,\mathrm{d}x & = & 0 \quad \forall y, \\ \int_y \tilde{k}^2(x, y)\,\mathrm{d}y & = & 0 \quad \forall x. \end{cases} \tag{4.26}$$

The uniqueness of this decomposition is proved in [37].

[2] Actually, this is shown for normal equations and with Dirichlet conditions, but in numerical experiments this result appears to hold also for Bi-CGSTAB and GMRES as iterative methods and using Sommerfeld absorbing conditions (S-ABC).

Now, the SoV preconditioner can be written as

$$\mathbf{M}_{SoV} = -\mathbf{B} - \hat{k}^2(\mathbf{x})\mathbf{I}, \tag{4.27}$$

where $\hat{k}^2(\mathbf{x}) = k_x^2(x) + k_y^2(y)$ is the '*SoV wavenumber*', i.e., $\tilde{k}(x, y) = 0$ is assumed in the preconditioner. Moreover, the SoV preconditioner applies *averaged* values of each absorbing boundary resulting in Dirichlet conditions. In other words, instead of absorbing boundary conditions, as in (4.18), we use the following condition for each boundary in the SoV preconditioner:

$$\frac{1}{L}\int_{m=0}^{L} \left(\frac{\partial}{\partial n} p(\mathbf{x}) + ikp(\mathbf{x}) \right) \, \mathrm{d}m = 0, \tag{4.28}$$

where m is the positive *right-angle* direction with respect to the outward normal n.

In simple smooth models with relatively low wavenumbers, the convergence rate of Bi-CGSTAB in combination with the SoV preconditioner is satisfactory. However, for complex models, the method is not good enough by increasing the wavenumbers, see [37].

In Chapter 6, we consider the SoV preconditioner in more detail and we investigate some properties of this preconditioner. Furthermore, we show how $\mathbf{M}_{SoV}\mathbf{x} = \mathbf{b}$ can be solved at low cost.

Chapter 5

Boundary Conditions

As known in the literature (see e.g. Boyce & DiPrima [11]), the solution of each boundary value problem depends on the choice of boundary conditions. Moreover, varying these conditions leads also to different eigenvalues and iterative behavior, in general. From this point of view, we have to pay attention to boundary conditions. In this chapter, three different aspects are considered which are related to boundary conditions.

In Chapter 2, we have seen that absorbing boundary conditions (ABC) are used in the HP. However, in research, one applies Dirichlet boundary conditions (DBC) for simplicity, leading to a real-valued problem. If there are no imaginary components in the main problem and in the preconditioner, then the eigenvalues of both the original and the preconditioned system are usually real-valued. We deal with this aspect in the first section of this chapter, where the wedge model is taken as test model.

Erlangga, Vuik & Oosterlee [15, 52] have shown the good results of the CSL preconditioner in the case of Dirichlet and Sommerfeld absorbing conditions (DBC respectively S-ABC). In Section 5.2, the HP, which applies conjugate Sommerfeld absorbing conditions (CS-ABC)), are considered and the performance of the CSL preconditioner is investigated in this problem.

Section 5.3 deals with the SoV preconditioner. In SoV, we have taken averaged values at the boundaries (see Subsection 4.3.2), but it could be more favorable to take other variants like maximum or minimum values instead of averaged values.

5.1 Comparison of Eigenvalues in the HP with CS-ABC and DBC

The spectra of both the SoV and CSL preconditioned system are examined in the following subsections.

5.1.1 SoV Preconditioned System

In this subsection, we investigate the spectra of both the original and the SoV preconditioned system in the HP with *absorbing* boundary conditions (CS-

ABC) and *Dirichlet* boundary conditions (DBC).

First, we start with giving and comparing convergence results of the iterative method for a few test problems and with varying gridsizes. Thereafter, the eigenvalue distribution of the HP is considered, using test problems (R) with respect to the wedge model. To restrict the computational work, we take $M, N = 25$ as gridsizes in this case.

Comparison Results using CS-ABC and DBC

In Table 5.1, the results of the convergence of the iterative method, using the SoV preconditioner with first CS-ABC and second DBC, can be found.

Test Problem	$M = N$	CS-ABC	DBC
(R)	25	6	6
	35	6	6
	45	7	6
(V)	25	35	57
	35	32	57
	45	34	46

Table 5.1: ***Number of iterations of Bi-CGSTAB with SoV preconditioner in test problems with absorbing conditions (second column) and with Dirichlet conditions (third column) in the wedge model.***

In the case of test problem (R), the results between CS-ABC and DBC are almost the same, whereas considerable differences can be observed in test problem (V). The number of points in (V) is not sufficiently large, as earlier mentioned in Section 2.9. Therefore, conclusions can not be made considering the results of (V) in Table 5.1. We expect that, if sufficient gridpoints are taken, the results applying both CS-ABC and DBC will be approximately the same.

Eigenvalues of the HP with CS-ABC

The plots of the eigenvalues of the original system $\mathbf{A}$ and of the preconditioned system $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ in test problem (R) can be found in Figure 5.1.

In both subplots of in Figure 5.1, we obtain *complex* eigenvalues. Moreover, it can be noticed that the range of the eigenvalues in the original system are much larger than in the preconditioned system and moreover, the eigenvalues of the preconditioned system are clustered around the coordinates (1,0). Therefore, comparing to the method without preconditioner, the preconditioned system usually leads to a better convergence performance.

Eigenvalues of the HP with DBC

Now, instead of taking absorbing conditions, we take Dirichlet conditions of the form:

$$p(\mathbf{x}) = 0, \quad \mathbf{x} \in \partial D, \tag{5.1}$$

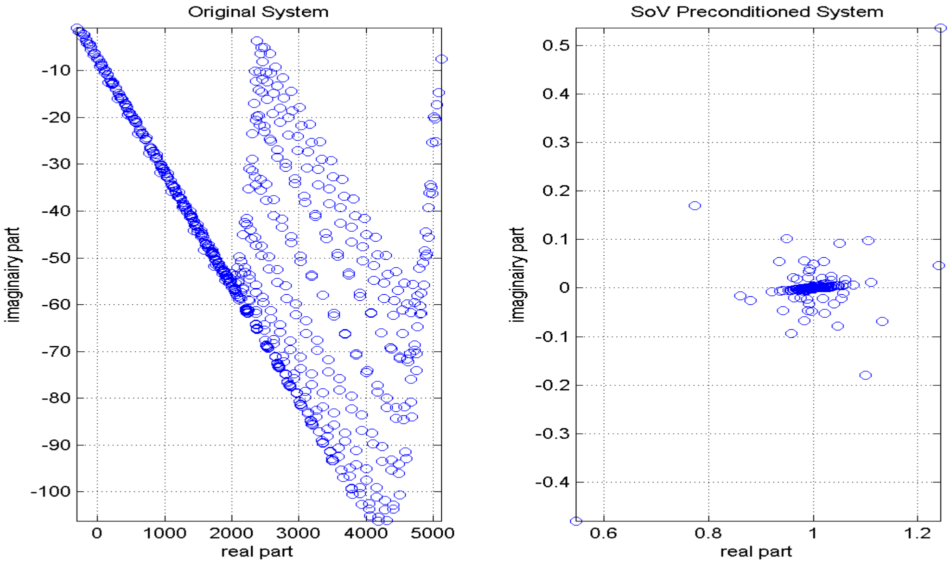


Figure 5.1: ***Eigenvalue distribution of the original system (left subplot) and of the SoV preconditioned system (right subplot) in test problem (R) with the wedge model and CS-ABC using*** $M, N = 25$***.***

in the HP. In this case, there are no imaginary components in the problem. Therefore, we expect the eigenvalues of both original and preconditioned system are real-valued. The results of the spectral plots can be seen in Figure 5.2.

Suprisingly, we observe in Figure 5.2 that the eigenvalues of the SoV preconditioned system are *complex*, while real-valued eigenvalues are obtained in the original system $\mathbf{A}$. A possible explanation can be found below.

Explanation of Complex Eigenvalues

One expects in advance that the Dirichlet problem, as defined above, gives real-valued eigenvalues of $\mathbf{M}_{SoV}^{-1}\mathbf{A}$. However, we have found eigenvalues which are complex-valued. This can be explained in the following way.

First, we make the observation that $\mathbf{A}$ and $\mathbf{M}_{SoV}$ are symmetric and real, due to the fact that Dirichlet instead of absorbing conditions have been used. Therefore, $\mathbf{M}_{SoV}^{-1}$ is also symmetric due to Lemma 3.2.

Unfortunately, $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ is not *symmetric*, in general. However, if we assume $\mathbf{A}$ to be *positive semi definite* (PSD), then we *do* have a matrix $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ with *real-valued* eigenvalues, see Theorem 5.1.

Theorem 5.1 *Let* $\boldsymbol{A}$ *and* $\boldsymbol{M}$ *to be both invertible matrices. Moreover, let* $\boldsymbol{A}$ *to be also PSD. Then* $\boldsymbol{M}^{-1}\boldsymbol{A}$ *consists of eigenvalues which are all real.*

Proof. This is an immediate consequence of Theorem 3.9.

□

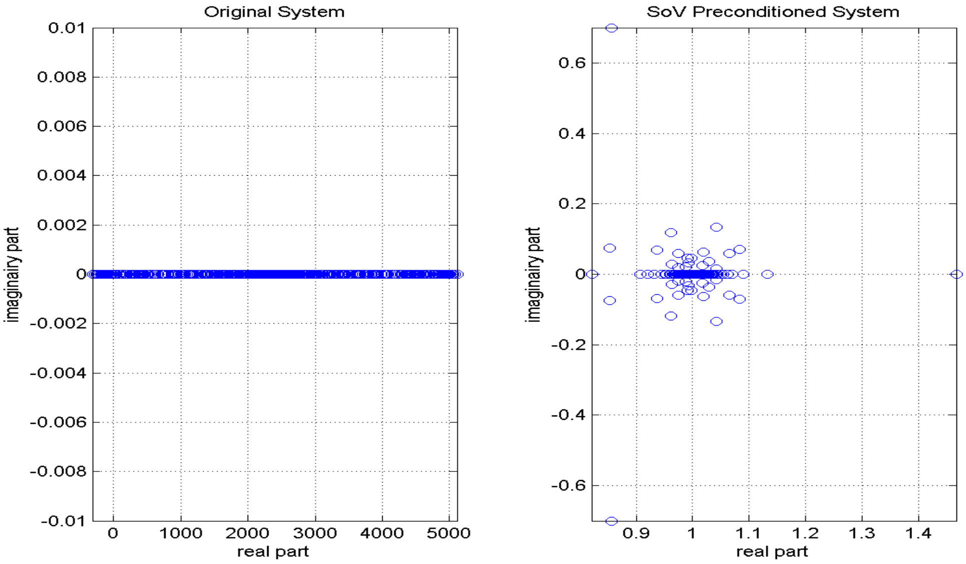


Figure 5.2: Eigenvalue distribution of the original system (left subplot) and of the SoV preconditioned system (right subplot) in test problem (R) with Dirichlet boundary conditions and the wedge model using $M, N = 25$.

Apparently, matrix **A** is not PSD in our problem, which is the result of the choices of

(a) the number of grid elements $M \times N$;

(b) the wavenumbers k_1 and k_2.

In our numerical experiments, we see that the system **A** becomes strongly indefinite (i.e., the range of eigenvalues becomes larger) when we increase (a) and/or (b). If we assume that the HP's with CS-ABC and DBC give approximately the same eigenvalues, then this can be motivated by using (1-dimensional) an analytical analysis, see **Appendix D**.

We conclude that only for sufficient small wavenumbers or sufficient small sizes of the domain, matrix **A** will be PSD.

Remark

If we assume a constant wavenumber in the HP, i.e., if we assume $k_1 = k_2$, then the SoV preconditioner is almost *exact*, since the equality $\tilde{k}(x, y) = 0$ holds. In other words:

$$\mathbf{M}_{SoV}^{-1}\mathbf{A} \approx \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}. \tag{5.2}$$

As a consequence, $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ is almost symmetric and has all eigenvalues near 1. Note that the averaged boundary conditions in SoV causes sligthly differences between **A** and $\mathbf{M}_{SoV}$ and, therefore, we use '≈' in (5.2) instead of equality ('=').

Conclusion

We have seen that for sufficient small wavenumber and/or sufficient small sizes of the domain, $\mathbf{A}$ is PSD or $\mathbf{M}_{SoV}^{-1}$ is approximately exact. Therefore, only in these cases $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ has real-valued eigenvalues.

We conclude that it does not make sense to use the Dirichlet problem in further research, since the eigenvalues of the preconditioned system are *not* all real-valued, which is favorable in spectral analysis.

5.1.2 CSL Preconditioned System

It is obvious that the CSL preconditioned system in the *Dirichlet* problem has complex eigenvalues, since $\mathbf{M}_{CSL}$ is *complex* due to

$$\mathcal{L}_{CSL} = -\Delta + (\alpha + \beta i)k^2, \tag{5.3}$$

where $\alpha = 0$ and $\beta = 1$ are taken.

If we choose the Shifted Laplace (SL) preconditioner to be *real*-valued and PSD, i.e., if we assume that $\alpha \geq 0$ and $\beta = 0$, it can be interesting to investigate the resulting SL-system $\mathbf{M}_{SL}^{-1}\mathbf{A}$ and look whether it has real-valued or complex eigenvalues. We find the following result. Since $\mathbf{M}_{SL}^{-1}$ is PSD, Theorem 5.1 is applicable. An immediate consequence is that all eigenvalues are *real*-valued in the SL preconditioned system $\mathbf{M}_{SL}^{-1}\mathbf{A}$ with $\alpha \geq 0$, in contrast to the SoV preconditioned system $\mathbf{M}_{SoV}^{-1}\mathbf{A}$!

5.2 Optimal CSL Preconditioner in HP

In Section 4.3.1, we have mentioned that the CSL preconditioner shows better performance than the SL preconditioners of Laird [29] and Bayliss *et al.* [6].

Define $\mathbf{M}_{\alpha,\beta}$ as follows:

$$\mathbf{M}_{\alpha,\beta} := \mathbf{M}_{SL} = -\mathbf{B} + (\alpha + \beta i)k^2(\mathbf{x}), \tag{5.4}$$

where $\alpha \geq 0$. Then, Erlangga, Vuik & Oosterlee [15, 52] have shown that $\mathbf{M}_{0,1}$ is the best preconditioner considering the HP with *Dirichlet* boundary conditions (DBC) using normal equations. Numerical experiments show also a good performance in the HP with *Sommerfeld* absorbing conditions (S-ABC).

However, in this thesis, we consider the HP with conjugate Sommerfeld absorbing conditions (CS-ABC). The question is whether the preconditioner $\mathbf{M}_{0,1}$ is also optimal in this problem.

Intuitively, we expect that $\mathbf{M}_{0,-1}$ is a better preconditioner than $\mathbf{M}_{0,1}$ (both applying CS-ABC). Some numerical experiments are done to confirm our expectation, see Table 5.2. Test problems (C++) and (V) are unrealistic problems, due to the low number of grid points, but the aim of these problems is to emphasize clearly the differences between the preconditioners $\mathbf{M}_{0,1}$ and $\mathbf{M}_{0,-1}$.

In Table 5.2, it can be observed that columns 4 and 7 and also columns 5 and 8 are the same. To understand this, we need the following theorem.

		S-ABC			CS-ABC		
Test Problem	M, N	$\mathbf{M}_{SoV}$	$\mathbf{M}_{0,1}$	$\mathbf{M}_{0,-1}$	$\mathbf{M}_{SoV}$	$\mathbf{M}_{0,1}$	$\mathbf{M}_{0,-1}$
(C++)	15	1	62	106	1	106	62
	25	1	60	114	1	114	60
	35	1	60	103	1	103	60
(V)	15	73	242	> 1000	73	> 1000	242
	25	35	159	> 1000	35	> 1000	159
	35	33	234	> 1000	33	> 1000	234

Table 5.2: Number of iterations applying Bi-CGSTAB with both SoV and CSL preconditioners in the constant and wedge model using Sommerfeld and conjugate Sommerfeld boundary conditions in both the original and preconditioner system.

Theorem 5.2 *Denote the matrix $\boldsymbol{A}$ obtained with S-ABC by $\boldsymbol{A}_{So}$ and CS-ABC by $\boldsymbol{A}_{CS}$. Then the real parts of the eigenvalues of the systems $\boldsymbol{M}_{0,1}^{-1}\boldsymbol{A}_S$ and $\boldsymbol{M}_{0,-1}^{-1}\boldsymbol{A}_{CS}$ are the same, where S-ABC is used in $\boldsymbol{M}_{0,1}^{-1}$ and CS-ABC is used in $\boldsymbol{M}_{0,-1}^{-1}$.*

Proof. Notice first that

$$\mathbf{A}_{CS} = \overline{\mathbf{A}_S}, \tag{5.5}$$

and

$$\mathbf{M}_{0,1}^{-1} = \overline{\mathbf{M}_{0,-1}^{-1}}. \tag{5.6}$$

Then

$$\mathbf{M}_{0,1}^{-1}\mathbf{A}_S = \overline{\mathbf{M}_{0,-1}^{-1}} \cdot \overline{\mathbf{A}_{CS}} = \overline{\mathbf{M}_{0,-1}^{-1}\mathbf{A}_{CS}}. \tag{5.7}$$

Using Lemma 3.10, we obtain that the real parts of the eigenvalues of $\mathbf{M}_{0,1}^{-1}\mathbf{A}_S$ and $\mathbf{M}_{0,-1}^{-1}\mathbf{A}_{CS}$ are the same.

□

We know that since the Helmholtz's equation is real-valued, only the boundary conditions leads to the imaginary components in the problem, resulting in a complex-valued matrix $\mathbf{A}$.

Consider now the preconditioned system using $\mathbf{M}_{0,1}$ in the HP with S-ABC. Applying Theorem 5.2, we derive that this system leads to the conjugate (and thus to the same real parts) of the eigenvalues as the preconditioned system using $\mathbf{M}_{0,-1}$ in the HP with CS-ABC. Next, we obtain the same results in columns 4 and 7 and also in columns 5 and 8 of Table 5.2 due to Conjecture 5.1.

Conjecture 5.1 *Let $\boldsymbol{A}_1\boldsymbol{p} = \boldsymbol{f}$ and $\boldsymbol{A}_2\boldsymbol{p} = \boldsymbol{f}$ be two linear systems where $\overline{\boldsymbol{A}_1} = \boldsymbol{A}_2$ and $\boldsymbol{f}$ an arbitrary vector. Then both systems requires (approximately) the same number of iterations using Bi-CGSTAB to solve $\boldsymbol{p}$.*

The results in Table 5.2 confirm Conjecture 5.1.

Thus, since $\mathbf{M}_{0,1}^{-1}$ is the best preconditioner in the case of $\mathbf{A}_S$, we obtain that $\mathbf{M}_{0,-1}^{-1}$ is the best preconditioner in the case of $\mathbf{A}_{CS}$. Therefore, in the remaining of this thesis we apply $\mathbf{M}_{0,-1}$ instead of $\mathbf{M}_{0,1}$ in CSL, since the HP deals with CS-ABC, after all.

5.3 Approximated Boundary Conditions in SoV

We have seen that the preconditioner $\mathbf{M}_{SoV}$ can be constructed in the following chronological way:

(i) the wavenumber $k(x, y)$ is decomposed into three parts, where the third part is the matrix $\tilde{k}(x, y)$;

(ii) averaged values of γ at the boundaries are computed. In practice, these are computed by summing the values along each boundary and dividing by the number of points;

(iii) the eigenvalues Λ and eigenfunctions $\mathbf{W}_R$ and $\mathbf{W}_L$ are computed.

Our point of concern in this section is the second step. The SoV-preconditioner takes averaged values at the boundaries, but the question is whether this the best choice. In the next subsection, we choose other values at the boundaries and investigate the performance of the preconditioner with these alternatives.

5.3.1 Approximations for the Boundaries in SoV

In the original SoV preconditioner [37], averaged values are applied at the boundaries. We try two other choices: SoV+ and SoV–. In SoV–, we take the *minimum* instead of the averaged value at each boundary and analogously, we apply the maximum value of each boundary in the SoV+. [1]

Now, some results of the variants of SoV are given in Table 5.3.

Test Problem	M, N	SoV	SoV+	SoV–
(R)	25	6	6	6
	35	6	6	6
(V)	25	35	35	34
	35	32	32	32
	45	34	34	34
(V++)	45	71	71	74

Table 5.3: Number of iterations of Bi-CGSTAB for three variants of SoV.

Considering Table 5.3, we conclude that the three variants of SoV show approximately the same iterative behavior. Hence, there is no preference to apply one of these variants in the preconditioner. In the following of this thesis, we use simply the original SoV with averaged values at the boundaries.

[1] For instance, in the layer model k_1 and k_2 are taken at the lower and upper boundary, respectively. Moreover, $\frac{k_1+k_2}{2}, \max(k_1, k_2), \min(k_1, k_2)$ are taken at both the right and left boundary in SoV, SoV+ and SoV–, respectively.

Chapter 6

Results of Test Problems

In the previous chapter, we have defined the preconditioners $\mathbf{M}_{SoV}$ and $\mathbf{M}_{CSL}$. Now, we are able to do some test runs in MATLAB to investigate the performance of these preconditioners. In Section 2.9, we have defined some test problems, which are used in our test runs.

The models for the wavenumber, as defined in Section 2.7, are applied in the test runs, where we pay extra attention to the wedge model.

6.1 Constant Model

We start with the constant model. Some results of the test runs can be found in Table 6.1. [1]

Considering Table 6.1, the following observations can be made:

- the SoV preconditioner is exact, since k is constant in the whole domain. Therefore, the iterative methods converge in one iteration using SoV;
- the CSL preconditioner shows good results, comparing to the case without preconditioner (column 3). However, differences in iterative behavior between SoV and CSL can be seen;
- results of GCR are omitted Table 6.1, because GCR and GMRES show approximately the same convergence behavior. This is the consequence of the reasons mentioned in Subsection 4.2.3. The difference between GMRES and GCR in number of iterations is equal or less than 2, in our test runs;
- In the original system without preconditioning, Bi-CGSTAB needs more iterations than GMRES, while GMRES and Bi-CGSTAB require approximately the same computational time, in general. This is not suprising, since GMRES applies *long* recurrences (see lines 5–8 in Algorithm 4.2) in contrast to Bi-CGSTAB, where short recurrences are used;

[1] Note that the standard IC preconditioner is only defined for real-valued matrices (see **Appendix C**). Therefore, we have taken the *real part* of the complex $\mathbf{A}$ in this preconditioner. Moreover, the IC preconditioner does not work in test problem (C++), because matrix $\mathbf{A}$ lost its positive definiteness during the Cholesky factorization in this case.

Bi-CGSTAB						
Test Problem	M, N	-	D	IC	CSL	SoV
(C)	25	88	94	77	16	1
	35	120	112	92	16	1
	45	189	182	162	16	1
(C++)	25	170	197	-	59	1
	35	199	200	-	57	1
	45	198	196	-	60	1

GMRES						
Test Problem	M, N	-	D	IC	CSL	SoV
(C)	25	54	54	47	19	1
	35	76	76	59	20	1
	45	97	97	73	20	1
(C++)	25	81	83	-	54	1
	35	112	112	-	53	1
	45	144	144	-	54	1

Table 6.1: Number of required iterations solving $Ap = f$ using iterative methods (GMRES, GCR and Bi-CGSTAB) without preconditioner (-), with the diagonal preconditioner (D), with the incomplete Cholesky preconditioner (IC), respectively, comparing to preconditioner based on complex shifted Laplace operator (CSL) and based on the separation–of–variables preconditioner (SoV).

- the effect of the diagonal preconditioner is minimal. This can be explained by looking at the eigenvalues. If we denote K_1 by the range of the eigenvalues (i.e., $|\lambda_{\max} - \lambda_{\min}|$) of $\mathbf{A}$ or $\mathbf{D}^{-1}\mathbf{A}\mathbf{D}^{-1}$ where $\mathbf{D}$ is the diagonal preconditioner, then we obtain the following results for $M, N = 45$:

  $$K_1(\mathbf{A}) = 1.57 \times 10^4, \quad K_1(\mathbf{D}^{-1}\mathbf{A}\mathbf{D}^{-1}) = 2.16. \tag{6.1}$$

  Moreover:

  $$K_2(\mathbf{A}) = 802.9, \quad K_2(\mathbf{D}^{-1}\mathbf{A}\mathbf{D}^{-1}) = 805.1, \tag{6.2}$$

  where K_2 denotes $|\lambda_{\max}|/|\lambda_{\min}|$. Thus, K_1 is in the preconditioned system much better, while the condition number K_2 is in both situation the same. Therefore, it leads to almost the same convergence rate in both methods.

  In **Appendix C** we have seen that the diagonal preconditioner makes sense if $\mathbf{A}$ is real-symmetric. Apparently, the diagonal preconditioner is not attractive for *complex-symmetric* matrices;

- note that, generally, increasing M, N leads to more iterations of Bi-CGSTAB and GMRES without preconditioner, which is the result of the stronger indefiniteness of matrix $\mathbf{A}$, i.e., increasing the gridsizes leads to a larger range of the eigenvalues of $\mathbf{A}$, and which is also the result of the increasing small eigenvalues near zero.

Due to the disappointing results for the D and IC preconditioners, we do not use these in further research. Thus, in the following sections, only SoV and CSL preconditioners are considered and analyzed.

Moreover, in the following sections we consider only the results obtained with Bi-CGSTAB and omit the results obtained with GMRES and GCR to restrict the computational work. Furthermore, Bi-CGSTAB does not have storage problems in practical applications due to the short recurrences, in contrast to the (non-restarted) GMRES. Therefore, we prefer Bi-CGSTAB rather than GMRES, as iterative method.

6.2 Rectangular Model

Some results of the test runs for the rectangular model can be found in Table 6.2.

Bi-CGSTAB			
Test Problem	M, N	CSL	SoV
(R)	25	50	3
	35	50	3
	45	52	3
(V)	25	138	4
	35	155	4
	45	154	3

Table 6.2: Number of required iterations solving the HP with the lrectangular model using Bi-CGSTAB in combination with the SoV and CSL preconditioners.

The SoV preconditioner is almost exact in this case. The few iterations, which are required in the iterative method, is caused by the averaged values at the boundaries in SoV. Apparently, it takes 3 or 4 iterations to get rid of these averaged values.

Moreover, it can be seen that SoV is again better than CSL in this rectangular model.

6.3 Wedge Model

In this section, we consider the wedge model. Some results of the test runs can be found in Table 6.3.

Below, some numerical analysis are done with respect to the solution, eigenvalues and convergence behavior.

Solution

For test problem (R), the solutions have been plotted from two points of view in Figure 6.1.

Bi-CGSTAB			
Test Problem	M, N	CSL	SoV
(R)	25	88	6
	35	90	6
	45	88	7
(R+)	25	55	11
	35	60	10
	45	57	10
(V)	25	160	35
	35	237	32
	45	271	34

Table 6.3: Number of required iterations solving the HP including the wedge model using Bi-CGSTAB in combination with the SoV and CSL preconditioners.

Eigenvalues

Moreover, the eigenvalue distribution of $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ in test problem (R) can be found in Figure 6.2 for gridsizes $M, N = 25, 35, 45$. In this figure, we observe that most eigenvalues lie around 1, resulting in fast convergence of the iterative method.

In Figure 6.2, it can be seen that the global structure of the eigenvalue distribution is more or less the same for varying M, N. In other words: increasing M and N leads to extra eigenvalues which lies in the cluster around 1. The number of eigenvalues outside this cluster looks to be more or less independent of the gridsizes.

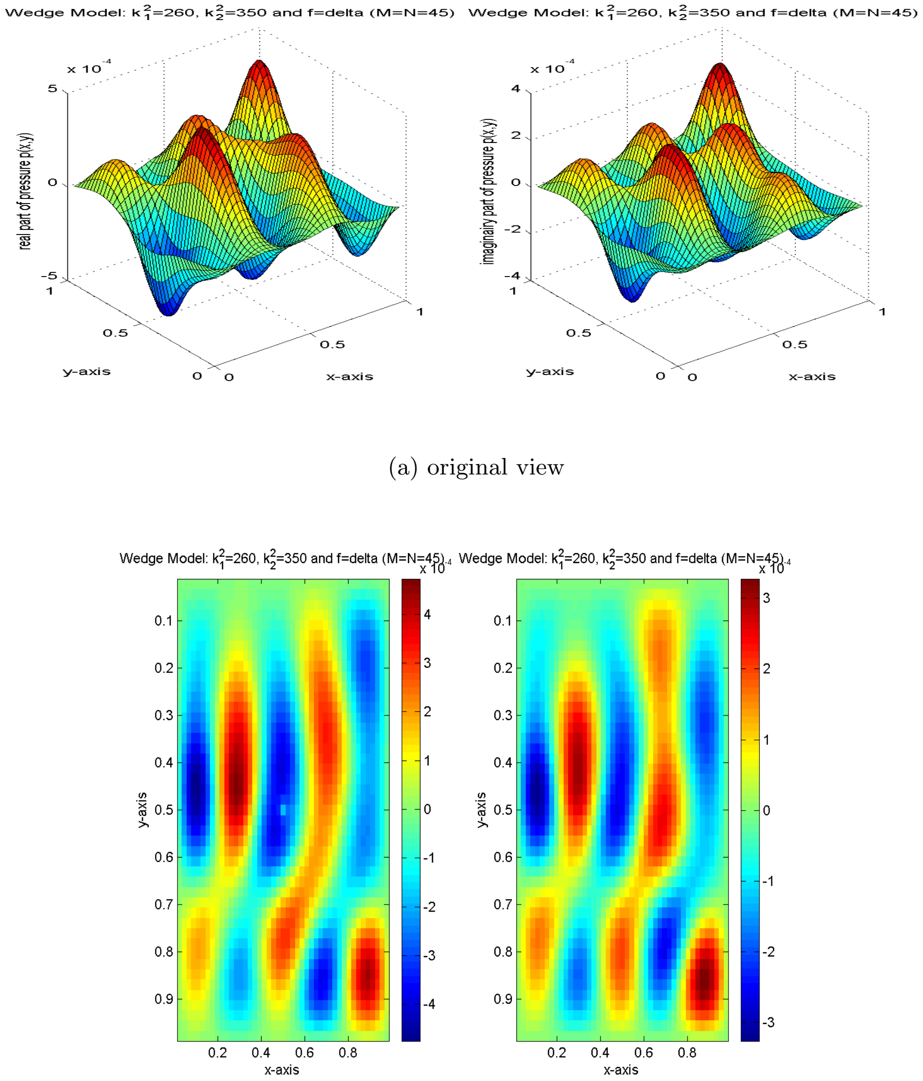
Next, the eigenvalue distribution of $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ in test problem (V) can be found in Figure 6.3. We note that, in fact, only a few eigenvalues near zero cause the relatively slow convergence for solving (V) iteratively. In the case with $M, N = 25$ (Figure 6.3(a)) there are two relatively large negative eigenvalues which results in the 'slow' convergence of 35 iterations. In the case of $M, N = 35$ (Figure 6.3(b)), one has exactly one 'bad' negative eigenvalues and, in total, three bad eigenvalues near zero. The best case is the case with $M, N = 45$ (Figure 6.3(c)), where all eigenvalues are positive, but, even in this case, it still requires 34 iterations to converge.

Residuals

We give the plots of the convergence behavior of test problem (R), see Figure 6.4. In each subplot, we have drawn the logarithm of the relative residual as in the RR criterium (see Subsection 4.2.4):

$$\frac{||\mathbf{f} - \mathbf{A}\mathbf{p}_j||}{||\mathbf{f}||}. \tag{6.3}$$

The erratic convergence behavior of the plots can be observed in Figure 6.4. Moreover, in all three cases the plots show *superlinear* behavior. These are known properties of the Bi-CGSTAB method.



(a) original view

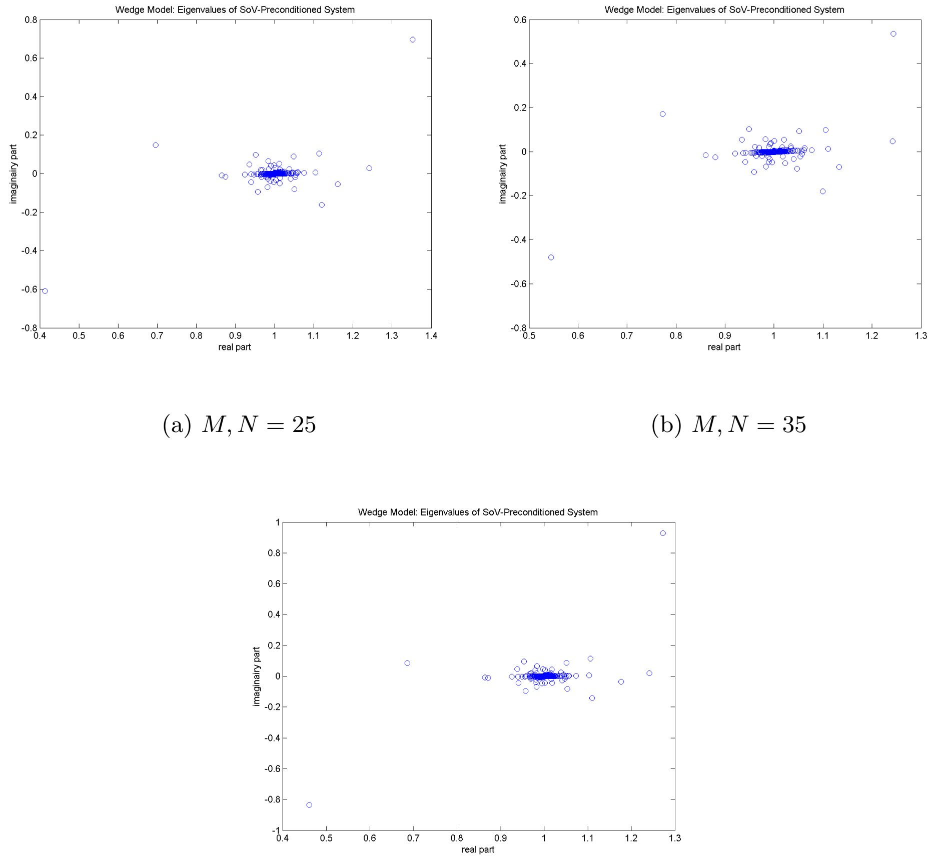
(b) top view

Figure 6.1: ***Solution p of the wedge model with*** $k_1^2 = 260, k_2^2 = 350$ ***on an unit domain for*** $M, N = 45$***. The real and imaginary parts of p are given in the left and the right subplots, respectively.***

Varying the Diagonal Interface

Next, the same problem with the wedge model is considered, but now with varying the diagonal interface between the layers. In Section 2.7, we have defined α and β to be real numbers satisfying $0 < \alpha < \beta < 1$, where it is assumed that the interface starts at $x = 0$ with αY and ends at $x = X$ with βY.

The choices $\alpha = 1/3$ and $\beta = 2/3$ are taken in the 'standard' wedge model.



(a) $M, N = 25$

(b) $M, N = 35$

(c) $M, N = 45$

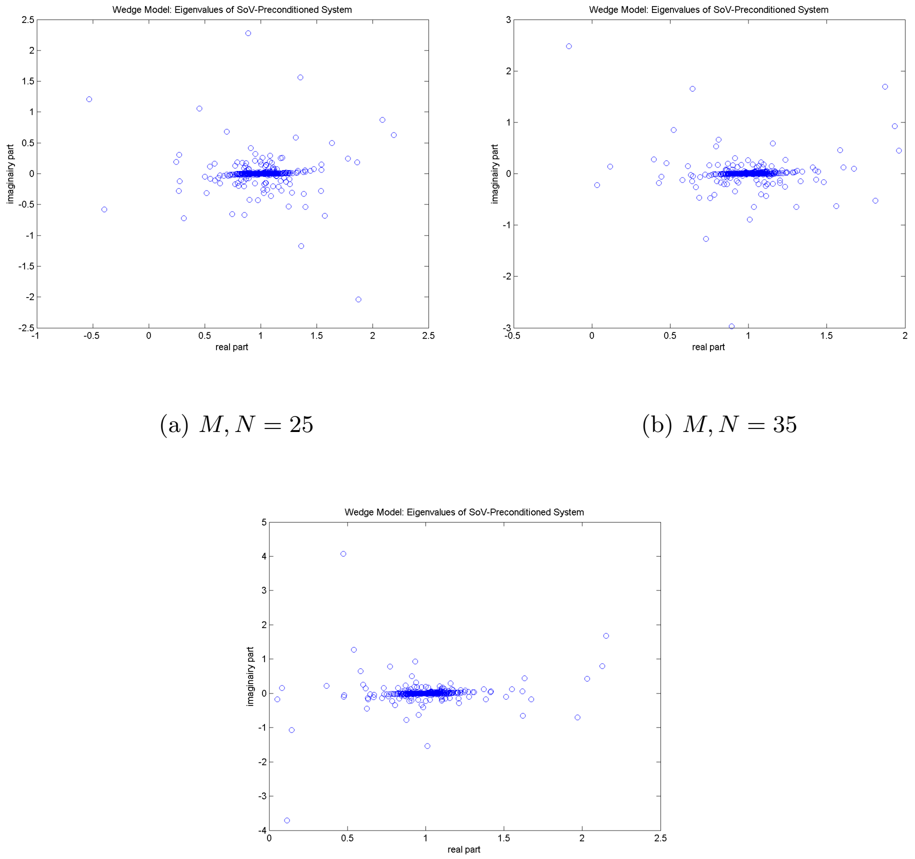
Figure 6.2: Eigenvalues of the system $\boldsymbol{M_{SoV}^{-1}A}$ ***with*** $k_1^2 = 260, k_2^2 = 350$ ***(test problem (R)) in the wedge model.***

We give the results for several other choices α and β, see Table 6.4, where we have applied $M, N = 25$ to reduce the computational time.

The first observation, which can be made, is that SoV is in all cases the better preconditioner relative to CSL. Moreover, we observe that the 'steeper' the diagonal interface the more SoV iterations are required, wheras CSL appears to be less dependent on the choice of the interface. This can be motivated by the fact that the problem becomes less separable, i.e., the term $\tilde{k}(x, y)$ becomes more important by steeping the diagonal boundary and hence, SoV requires more iterations to converge.

6.4 Other Models

We have seen the good performance of SoV in the previous models. Now, we consider the sinus, random and min-max model. We hope that, in these cases, CSL performs better than SoV, so that these models could be used for a 'better'



(a) $M, N = 25$

(b) $M, N = 35$

(c) $M, N = 45$

Figure 6.3: Eigenvalues of the system $M_{SoV}^{-1}A$ with $k_1^2 = 400, k_2^2 = 1200$ (test problem (V)) in the wedge model.

further analysis of the combined preconditioners in Chapters 8 and 9.

The results of test problem (R) can be found in Table 6.5. We observe that all models, using the SoV preconditioner, give fast convergence. Furthermore, SoV performs better than CSL in all cases, considering Table 6.5.

6.5 Conclusion

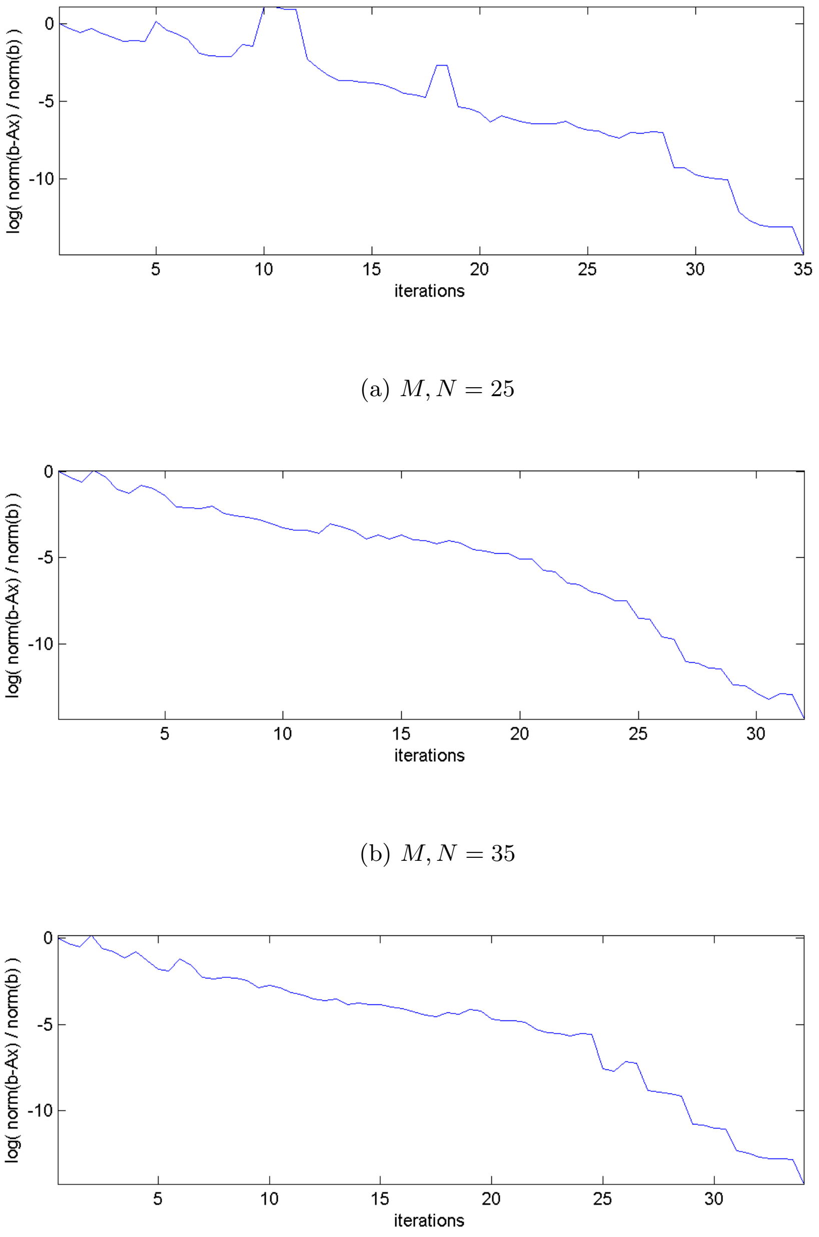
We conclude that the SoV preconditioner is always (much) better than the CSL preconditioner, in all test runs we have carried out in this chapter. Apparently, we have taken only relative *small* test problems. This observation becomes also clear, when we compare these to test problems and results of Mulder & Plessix [37]. For a good spectral analysis with respect to the failure of SoV in complex models and for an effective examination of the possibilities for combined preconditioners, *larger* test problems are needed with gridsizes $M, N > 45$. This is left for further research.

α	β	CSL	SoV
25/50	26/50	144	4
5/10	6/10	163	11
2/5	3/5	166	19
1/3	2/3	160	35
1/5	4/5	178	56
1/10	9/10	169	61
1/50	49/50	162	61

Table 6.4: Number of iterations of Bi-CGSTAB with SoV and CSL preconditioning in test problem (R) with the wedge model and different choices of α, β ***in the case*** $M, N = 25$.

		Sinus		**Random**		**Min–Max**	
Test Problem	M, N	SoV	CSL	SoV	CSL	SoV	CSL
(R)	25	7	80	20	128	7	80
(R)	35	7	84	4	85	6	83

Table 6.5: Iterative results of Bi-CGSTAB in test problem (R) with the sinus, random and min-max models, respectively.



(a) $M, N = 25$

(b) $M, N = 35$

(c) $M, N = 45$

Figure 6.4: Relative residuals of the system $\boldsymbol{M_{SoV}^{-1}A}$ ***with*** $k_1^2 = 400, k_2^2 = 1200$ ***(test problem (V)) in the wedge model.***

Chapter 7

Improving the SoV Preconditioner

In Chapter 4, we have seen that the preconditioner $\mathbf{M}_{SoV}$ based on separation of variables (Plessix & Mulder [37]), or briefly SoV(-preconditioner), can be used for Krylov iterative methods which solve the HP.

Since the wavenumber k depends on the spatial coordinates, it prevents us from using separation of the HP. However, k can be decomposed such that it is 'almost' separable and hence, this decomposition can be used as a preconditioner.

However, we have mentioned earlier in Subsection 4.3.2 that this SoV preconditioner fails in complex models or in models with high wavenumbers. In this chapter, we try to improve the SoV preconditioner such that it is also applicable in these models. Therefore, we have to consider the original SoV technique in more detail. This is done in Section 7.1.

In Section 7.2, we give a mathematical motivation why the SoV preconditioner fails in some situations by considering the decomposition of k.

We end with several attempts to construct an improved SoV preconditioner, in Section 7.3.

7.1 SoV Technique

In this section, we first give the decomposition of the wavenumber into $k_x^2(x)$, $k_y^2(y)$ and a remaining term $\tilde{k}^2(x, y)$. Thereafter, matrix $\mathbf{A}$ is also decomposed into three terms, such that the separation–of–variables technique makes sense. Then we show how $\mathbf{M}_{SoV}\mathbf{x} = \mathbf{b}$ can be computed efficiently with the decomposed $\mathbf{A}$ and k.

7.1.1 Decomposition of the Wavenumber

In the SoV preconditioner, we have to decompose the wavenumber $k(x, y)$ into three parts: a first part which only depends on x, a second part which only depends on y, and, finally, a remaining part which satisfies specific conditions as given in (4.25). In fact, $k(x, y)$ is divided in a separable and a non-separable part.

We assume in this section that $(x, y) \in \Omega$ where $(x, y) = [x_a, x_b] \times [y_a, y_b]$ with $0 < x_a < x_b$ and $0 < y_a < y_b$. Then, we take

$$\left\{ \begin{array}{ccccc} x_a & = & y_a & = & 0; \\ x_b & = & y_b & = & L, \end{array} \right. \tag{7.1}$$

with $L = 1$ since we deal with an unit domain in our HP.

Next, it appears to be always possible to decompose an arbitrary function $k(x, y)$ into three terms, where the third term satisfies the following conditions:

$$\left\{ \begin{array}{l} k^2(x,y) = k_x^2(x) + k_y^2(y) + \tilde{k}^2(x,y), \\ \\ \int_{x_a}^{x_b} \tilde{k}^2(x,y)\ \mathrm{d}x = 0, \quad \forall y, \\ \\ \int_{y_a}^{y_b} \tilde{k}^2(x,y)\ \mathrm{d}y = 0, \quad \forall x, \end{array} \right. \tag{7.2}$$

Then, a possible decomposition is

$$\left\{ \begin{array}{rcl} k_x^2(x) & = & \frac{1}{Y} \int_{y_a}^{y_b} k^2(x,y)\ \mathrm{d}y, \\ \\ k_y^2(y) & = & \frac{1}{X} \int_{x_a}^{x_b} k^2(x,y) - k_x^2(x)\ \mathrm{d}x, \\ \\ \tilde{k}^2(x,y) & = & k^2(x,y) - k_x^2(x) - k_y^2(y), \end{array} \right. \tag{7.3}$$

with

$$\left\{ \begin{array}{ccccc} X & = & \int_{x_a}^{x_b} 1\ \mathrm{d}x & = & x_b - x_a, \\ \\ Y & = & \int_{y_a}^{y_b} 1\ \mathrm{d}y & = & y_b - y_a, \end{array} \right. \tag{7.4}$$

where we have assumed that all integrals are finite. The above statement is proved in Theorem 7.1.

Theorem 7.1 *Let $k(x, y)$ be an arbitrary function. Define $k_x^2(x)$, $k_y^2(y)$ and $\tilde{k}^2(x, y)$ as in (7.3), where $\tilde{k}^2(x, y)$ is assumed to satisfy*

$$\int_{x_a}^{x_b} \tilde{k}^2(x,y)\ dx < \infty, \quad \int_{y_a}^{y_b} \tilde{k}^2(x,y)\ dy < \infty. \tag{7.5}$$

Then $k(x, y), k_x(x), k_y(y)$ and $\tilde{k}^2(x, y)$ satisfy

$$\left\{ \begin{array}{l} k^2(x,y) = k_x^2(x) + k_y^2(y) + \tilde{k}^2(x,y), \\ \\ \int_{x_a}^{x_b} \tilde{k}^2(x,y)\ dx = 0, \quad \forall y, \\ \\ \int_{y_a}^{y_b} \tilde{k}^2(x,y)\ dy = 0, \quad \forall x. \end{array} \right. \tag{7.6}$$

Proof. We define first

$$k_0^2 = \frac{1}{XY} \int_{y_a}^{y_b} \int_{x_a}^{x_b} k^2(x,y) \ \mathrm{d}x \ \mathrm{d}y. \tag{7.7}$$

The *first* expression of (7.6) is satisfied by construction of (7.3).

The *second* expression of (7.6) is true, since

$$\begin{aligned}
\textstyle\int_{x_a}^{x_b} \ \tilde{k}^2(x,y) \ \mathrm{d}x &= \textstyle\int_{x_a}^{x_b} \ k^2(x,y) - k_x^2(x) - k_y^2(y) \ \mathrm{d}x \\
&= \textstyle\int_{x_a}^{x_b} \ k^2(x,y) - k_x^2(x) \ \mathrm{d}x - \int_{x_a}^{x_b} \ k_y^2(y) \ \mathrm{d}x \\
&= X k_y^2(y) - k_y^2(y) \textstyle\int_{x_a}^{x_b} \ 1 \ \mathrm{d}x \\
&= X k_y^2(y) - X k_y^2(y) \\
&= 0.
\end{aligned}$$

Next, we proof the *third* expression of (7.6). It yields

$$\begin{aligned}
\textstyle\int_{y_a}^{y_b} \ \tilde{k}^2(x,y) \ \mathrm{d}y &= \textstyle\int_{y_a}^{y_b} \ k^2(x,y) - k_x^2(x) - k_y^2(y) \ \mathrm{d}y \\
&= \textstyle\int_{y_a}^{y_b} \ k^2(x,y) \ \mathrm{d}y - \int_{y_a}^{y_b} \ k_x^2(x) \ \mathrm{d}y - \int_{y_a}^{y_b} \ k_y^2(y) \ \mathrm{d}y \\
&= Y k_x^2(x) - k_x^2(x) \textstyle\int_{y_a}^{y_b} \ 1 \ \mathrm{d}y - \underbrace{\int_{y_a}^{y_b} \ k_y^2(y) \ \mathrm{d}y}_{=0} \\
&= Y k_x^2(x) - Y k_x^2(x) \\
&= 0,
\end{aligned}$$

where we have used the fact that

$$\begin{aligned}
\textstyle\int_{y_a}^{y_b} \ k_y^2(y) \ \mathrm{d}y &= \textstyle\int_{y_a}^{y_b} \ \left(\frac{1}{X} \int_{x_a}^{x_b} \ k^2(x,y) - k_x^2(x) \ \mathrm{d}x \right) \ \mathrm{d}y \\
&= \textstyle\frac{1}{X} \int_{x_a}^{x_b} \ \int_{y_a}^{y_b} \ k^2(x,y) \ \mathrm{d}x \ \mathrm{d}y - \frac{1}{X} \int_{x_a}^{x_b} \ \int_{y_a}^{y_b} \ k_x^2(x) \ \mathrm{d}x \ \mathrm{d}y \\
&= Y k_0^2 - \textstyle\frac{1}{X} \int_{x_a}^{x_b} \ \int_{y_a}^{y_b} \ \left(\frac{1}{Y} \int_{y_a}^{y_b} \ k^2(x,y) \ \mathrm{d}y \right) \ \mathrm{d}x \ \mathrm{d}y \\
&= Y k_0^2 - \textstyle\frac{1}{XY} \int_{x_a}^{x_b} \ \int_{y_a}^{y_b} \ \int_{y_a}^{y_b} \ k^2(x,y) \ \mathrm{d}x \ \mathrm{d}y^2 \\
&= Y k_0^2 - \textstyle\int_{y_a}^{y_b} \ k_0^2 \ \mathrm{d}y \\
&= Y k_0^2 - k_0^2 \textstyle\int_{y_a}^{y_b} \ 1 \ \mathrm{d}y \\
&= Y k_0^2 - Y k_0^2 \\
&= 0.
\end{aligned}$$

□

In Example E.1 of **Appendix E**, we give an application of Theorem 7.1.

Numerical Implementation

Assume that **K** is given with elements

$$\mathbf{K}_{i+(j-1)N,\ i+(j-1)N} = k^2(x_i, y_j), \tag{7.8}$$

for all $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$. Since both k_x and k_y are only dependent in one direction, we can represent the $MN \times MN$ *diagonal* matrix **K** as follows:

$$\mathbf{K} = \mathbf{I}_y \otimes \mathbf{K}_x + \mathbf{K}_y \otimes \mathbf{I}_x + \widetilde{\mathbf{K}}, \tag{7.9}$$

where the Kronecker symbol ⊗, as defined in Section 3.1, is used. Next, denote

$$\Delta x = \frac{x_M - x_1}{M - 1}, \quad \Delta y = \frac{y_M - y_1}{N - 1}. \tag{7.10}$$

Then, using the rectangular rule (see e.g. Smith [45]) [1], the non-zero elements of diagonal matrices $\mathbf{K}_x$ and $\mathbf{K}_y$ can be determined as follows:

$$\begin{cases} (\mathbf{K}_x)_{p,p} &= \frac{1}{y_N - y_1} \sum_{n=1}^{N-1} \Delta y \mathbf{K}_{p+(n-1)N,\ p+(n-1)N}, \\ (\mathbf{K}_y)_{q,q} &= \frac{1}{x_M - x_1} \sum_{m=1}^{M-1} \Delta x \left(\mathbf{K}_{m+(q-1)N,\ m+(q-1)N} - (\mathbf{K}_x)_{m,m} \right), \end{cases} \tag{7.11}$$

for all $p = 1, \ldots, M$ and $q = 1, \ldots, N$. Using (7.10), we can rewrite Expression (7.11) as

$$\begin{cases} (\mathbf{K}_x)_{p,p} &= \frac{1}{N-1} \sum_{n=1}^{N-1} \mathbf{K}_{p+(n-1)N,\ p+(n-1)N}, \\ (\mathbf{K}_y)_{q,q} &= \frac{1}{M-1} \sum_{m=1}^{M-1} \left(\mathbf{K}_{m+(q-1)N,\ m+(q-1)N} - (\mathbf{K}_x)_{m,m} \right). \end{cases} \tag{7.12}$$

Subsequently, matrix $\widetilde{\mathbf{K}}$ has the following form:

$$\widetilde{\mathbf{K}} = \mathbf{K} - \mathbf{I}_y \otimes \mathbf{K}_x - \mathbf{K}_y \otimes \mathbf{I}_x. \tag{7.13}$$

Note that $\widetilde{\mathbf{K}}$ is also a *diagonal* matrix.

In **Appendix F**, the resulting plots, using the numerical implementation of k, are drawn for various models of the original and the SoV wavenumber.

[1] There are more accurate ways to approximate integrals, like the trapezoidal and Simpson's rule (see e.g. [2]), but they are not taken into account in this thesis.

7.1.2 Averaged Values at the Boundaries

In Section 2.5, we have seen that discretizing the HP gives us the linear system $\mathbf{Ap} = \mathbf{f}$. Matrix $\mathbf{A}$ has dimensions $MN \times MN$, where the main diagonal consists of the elements

$$\mathbf{A}(d,d) = \frac{2}{\Delta x^2} + \frac{2}{\Delta y^2} - k_{m,n}^2 + \gamma_{m,n}, \tag{7.14}$$

for $d = 1, 2, \dots, MN$. The coefficients $\gamma_{m,n}$ have been given by

$$\gamma_{m,n} = 0, \quad \text{for } m = 2, 3, \dots, M-1 \ \text{ and } \ n = 2, 3, \dots, N-1, \tag{7.15}$$

and

$$\gamma_{m,n} = \begin{cases} \gamma_x^{\min}(n) & = & \dfrac{1}{\Delta x^2(1 + ik_{0,n}\Delta x)} & \text{if } m = 1; \\[2ex] \gamma_x^{\max}(n) & = & \dfrac{1}{\Delta x^2(1 + ik_{M+1,n}\Delta x)} & \text{if } m = M; \\[2ex] \gamma_y^{\min}(m) & = & \dfrac{1}{\Delta y^2(1 + ik_{m,0}\Delta y)} & \text{if } n = 1; \\[2ex] \gamma_y^{\max}(m) & = & \dfrac{1}{\Delta y^2(1 + ik_{m,N+1}\Delta y)} & \text{if } n = N. \end{cases} \tag{7.16}$$

In the SoV preconditioner, we use *constant averaged approximations* for the absorbing conditions (7.16) of the form

$$\begin{cases} \gamma_x^{\min}(n) & = & \tilde{\gamma}_x^{\min}; \\[1ex] \gamma_x^{\max}(n) & = & \tilde{\gamma}_x^{\max}; \\[1ex] \gamma_y^{\min}(m) & = & \tilde{\gamma}_y^{\min}; \\[1ex] \gamma_y^{\max}(m) & = & \tilde{\gamma}_y^{\max}, \end{cases} \tag{7.17}$$

where $\tilde{\gamma}_x^{\min}, \tilde{\gamma}_x^{\max}, \tilde{\gamma}_y^{\min}, \tilde{\gamma}_y^{\max} \in \mathbb{C}$. Therefore, constant values of k are chosen at each boundary, which are the *avaraged* value along that boundary. In fact, the expressions, given in (7.17), are the discrete forms of the integral in Expression (4.28).

Constant values at the boundaries are needed in the SoV preconditioner, to ensure the decomposition of matrix $\mathbf{A}$, which are derived in the next section.

7.1.3 Approximation of A

The following decomposition of matrix $\mathbf{A}$ can be made:

$$\mathbf{A} = \mathbf{X} + \mathbf{Y} - \mathbf{K}, \tag{7.18}$$

where $\mathbf{X}, \mathbf{Y}$ are complex matrices due to $\tilde{\gamma}$ and furthermore, $\mathbf{K}$ is a real matrix with the same sizes as $\mathbf{A}$, see below. However, we have to use an approximation of $\mathbf{A}$, denoted by $\hat{\mathbf{A}}$, due to the SoV technique:

$$\hat{\mathbf{A}} = \mathbf{X} + \mathbf{Y} - \hat{\mathbf{K}}, \tag{7.19}$$

where $\hat{\mathbf{K}}$ is also a real matrix. Below we describe these matrices in more detail.

Matrix X

Using Section 2.5, matrix $\mathbf{X}$ can be constructed by

$$\begin{cases} \mathbf{X}(d,d) & = & \dfrac{2}{\Delta x^2} + \gamma_{m,n}; \\ \\ \mathbf{X}(d,d+1) & = & \mathbf{X}_1(d,d-1) & = & -\dfrac{1}{\Delta x^2}, \end{cases} \tag{7.20}$$

with $d = 1, 2, \ldots, MN$ and

$$\gamma_{m,n} = \begin{cases} \tilde{\gamma}_x^{\min} & \text{if } m = 1; \\ \tilde{\gamma}_x^{\max} & \text{if } m = M; \\ 0 & \text{otherwise.} \end{cases} \tag{7.21}$$

Therefore, $\mathbf{X}$ is a $MN \times MN$ block-diagonal matrix with $M \times M$ blocks, which are equal to each other. Using again the definition of the Kronecker product, we can write

$$\mathbf{X} = \mathbf{I}_y \otimes \mathbf{A}_x, \tag{7.22}$$

where $\mathbf{I}_y$ is the $N \times N$ identity matrix and $\mathbf{A}_x$ is an $M \times M$ matrix with the following non-zero elements:

$$\begin{cases} \mathbf{A}_x(m,m) & = & \dfrac{2}{\Delta x^2} + \begin{cases} \tilde{\gamma}_x^{\min} & \text{if } m = 1; \\ \tilde{\gamma}_x^{\max} & \text{if } m = M; \\ 0 & \text{otherwise,} \end{cases} \\ \\ \mathbf{A}_x(m+1,m) & = & \mathbf{A}_x(m,m+1) = -\dfrac{1}{\Delta x^2}, \end{cases} \tag{7.23}$$

for $m = 1, \ldots, M$. Note that Expression (7.22) can be represented by

$$\mathbf{I}_y \otimes \mathbf{A}_x = \begin{bmatrix} \mathbf{A}_x & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{A}_x & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} = \mathbf{X}, \tag{7.24}$$

which consists of N blocks of $\mathbf{A}_x$.

Matrix Y

Matrix $\mathbf{Y}$ can be constructed by

$$\begin{cases} \mathbf{Y}(d,d) & = & \dfrac{2}{\Delta y^2} + \gamma_{m,n}; \\ \\ \mathbf{Y}(d,d+M) & = & \mathbf{A}_1(d,d-M) & = & -\dfrac{1}{\Delta y^2}, \end{cases} \tag{7.25}$$

with $d = 1, 2, \ldots, MN$ and

$$\gamma_{m,n} = \begin{cases} \tilde{\gamma}_y^{\min} & \text{if } n = 1; \\ \tilde{\gamma}_y^{\max} & \text{if } n = N; \\ 0 & \text{otherwise.} \end{cases} \tag{7.26}$$

This is an $MN \times MN$ tri-block-diagonal matrix of the form

$$\mathbf{Y} = \begin{bmatrix} \mathbf{D}_1 & \mathbf{L} & & & \\ \mathbf{L} & \mathbf{D}_2 & \mathbf{L} & & \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{L} & \mathbf{D}_2 & \mathbf{L} \\ & & & \mathbf{L} & \mathbf{D}_3 \end{bmatrix}, \tag{7.27}$$

where $\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3, \mathbf{L}$ are diagonal submatrices with dimensions $M \times M$. Note that the non-zero elements at the diagonal of each of these submatrices are the same. Now, we can write matrix $\mathbf{Y}$ as

$$\mathbf{Y} = \mathbf{A}_y \otimes \mathbf{I}_x, \tag{7.28}$$

where $\mathbf{I}_x$ is the $M \times M$ identity matrix and $\mathbf{A}_y$ is an $N \times N$ matrix with the following non-zero elements:

$$\begin{cases} \mathbf{A}_y(n,n) & = \dfrac{2}{\Delta y^2} + \begin{cases} \tilde{\gamma}_y^{\min} & \text{if } n = 1; \\ \tilde{\gamma}_y^{\max} & \text{if } n = N; \\ 0 & \text{otherwise,} \end{cases} \\ \\ \mathbf{A}_y(n+1,n) & = \mathbf{A}_y(n,n+1) = -\dfrac{1}{\Delta y^2}, \end{cases} \tag{7.29}$$

for $n = 1, \ldots, N$. Note that expression (7.28) can be written as

$$\begin{aligned} \mathbf{Y} = \mathbf{A}_y \otimes \mathbf{I}_x &= \begin{bmatrix} (a_y)_{11}\mathbf{I}_x & (a_y)_{12}\mathbf{I}_x & & \\ (a_y)_{21}\mathbf{I}_x & (a_y)_{22}\mathbf{I}_x & (a_y)_{23}\mathbf{I}_x & \\ & \ddots & \ddots & \ddots \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{D}_1 & \mathbf{L} & & \\ \mathbf{L} & \mathbf{D}_2 & \mathbf{L} & \\ & \ddots & \ddots & \ddots \end{bmatrix}, \end{aligned} \tag{7.30}$$

where $(a_y)_{ij}$, for all $i, j = 1, 2, \ldots, N$, are the elements of $\mathbf{A}_y$.

Matrix $\hat{\mathbf{K}}$

Diagonal matrix $\mathbf{K}$ of size $MN \times MN$ can be represented by

$$\text{diag}(\mathbf{K}(m + (n-1)N) = k_{m,n}^2, \tag{7.31}$$

with $m = 1, 2, \ldots, M$ and $n = 1, 2, \ldots, N$.

If Ω is taken as in a *rectangular model*, then the wavenumber $k(x,y)$ is constant in x-direction and variable in y-direction. In this model, we can construct easily the decomposition

$$k^2(x,y) = k_y^2(y). \tag{7.32}$$

In general, we have a more complicated model, like the wedge of sinus model (see Section 2.7). In this case, matrix $\mathbf{K}$ prevents us from using separation of variables. However, Theorem 7.1 shows that the square of the wavenumber can be uniquely decomposed into

$$k^2(x,y) = k_x^2(x) + k_y^2(y) + \tilde{k}^2(x,y), \tag{7.33}$$

which results in the diagonal matrix $\mathbf{K}$. In the SoV preconditioner, we neglect $\tilde{k}(x,y)$ in (7.33), leading to a modified wavenumber $\hat{k}$, i.e.,

$$\hat{k}^2(x,y) = k_x^2(x) + k_y^2(y). \tag{7.34}$$

The quantity $\hat{k}$ is called the *SoV wavenumber*. In discretized form we can express (7.34) as

$$(\hat{k}^2)_{m,n} = (k_x^2)_m + (k_y^2)_n, \tag{7.35}$$

for all $m = 1, 2, \ldots, M$ and $n = 1, 2, \ldots, N$. Then, matrix $\hat{\mathbf{K}}$ becomes

$$\hat{\mathbf{K}} = \mathbf{X}_{\hat{\mathbf{K}}} + \mathbf{Y}_{\hat{\mathbf{K}}}, \tag{7.36}$$

where the definitions of $\mathbf{X}_{\hat{\mathbf{K}}}$ and $\mathbf{Y}_{\hat{\mathbf{K}}}$ can be found below.

Matrices $\mathbf{X}_{\hat{\mathbf{K}}}$ and $\mathbf{Y}_{\hat{\mathbf{K}}}$

Diagonal matrices $\mathbf{X}_{\hat{\mathbf{K}}}$ and $\mathbf{Y}_{\hat{\mathbf{K}}}$ can be represented by

$$\mathbf{X}_{\hat{\mathbf{K}}} = \mathbf{I}_y \otimes \mathbf{K}_x, \tag{7.37}$$

and

$$\mathbf{Y}_{\hat{\mathbf{K}}} = \mathbf{K}_y \otimes \mathbf{I}_x, \tag{7.38}$$

where $\mathbf{K}_x$ and $\mathbf{K}_y$ are defined as in (7.12).

Matrix $\hat{\mathbf{A}}$

Using (7.22) and (7.28), matrix $\mathbf{A}$ turns out to be

$$\hat{\mathbf{A}} = \mathbf{I}_y \otimes \mathbf{A}_x + \mathbf{A}_y \otimes \mathbf{I}_x - \hat{\mathbf{K}}, \tag{7.39}$$

where we put a 'hat' at $\mathbf{A}$ to emphasize the fact that this is an approximation of the real $\mathbf{A}$, if k is non-separable. Moreover, matrix $\hat{\mathbf{K}}$ becomes

$$\hat{\mathbf{K}} = \mathbf{I}_y \otimes \mathbf{K}_x + \mathbf{K}_y \otimes \mathbf{I}_x, \tag{7.40}$$

using Expressions (7.37) and (7.38). This leads to

$$\hat{\mathbf{A}} = \mathbf{I}_y \otimes (\mathbf{A}_x - \mathbf{K}_x) + (\mathbf{A}_y - \mathbf{K}_y) \otimes \mathbf{I}_x. \tag{7.41}$$

Observe that matrices $\hat{\mathbf{A}}, \mathbf{A}_x - \mathbf{K}_x$ and $\mathbf{A}_y - \mathbf{K}_y$ all have symmetric real parts, which will be used later.

7.1.4 Transformation of $\hat{\mathbf{A}}$ into D

The SoV technique consists of replacing a problem of size MN by M 1-D problems of size N. Therefore, we need M subblocks of $\mathbf{D}_m$. In this subsection, we derive these subblocks.

Eigenvalue and Eigenvector Decomposition

The *eigenvector* and *eigenvalue decomposition* is required of $\mathbf{A}_x - \mathbf{K}_x$:

$$\mathbf{W}_L^H(\mathbf{A}_x - \mathbf{K}_x)\mathbf{W}_R = \Lambda, \tag{7.42}$$

with Λ a diagonal eigenvalue matrix, $\mathbf{W}_R$ a corresponding matrix of the right eigenvectors and $\mathbf{W}_L$ a corresponding matrix of the left eigenvectors of $\mathbf{A}_x - \mathbf{K}_x$. [2] Then, matrices $\mathbf{W}_L$ and $\mathbf{W}_R$ satisfy $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$, where $\mathbf{I}$ is the $MN \times MN$ identity matrix, see Theorem 3.1.

Matrix B

Matrix $\mathbf{I}_y \otimes \mathbf{W}_L^H$ can be written as

$$\mathbf{I}_y \otimes \mathbf{W}_L^H = \begin{bmatrix} \mathbf{W}_L^H & & & \\ & \mathbf{W}_L^H & & \\ & & \ddots & \\ & & & \mathbf{W}_L^H \end{bmatrix}, \tag{7.43}$$

and, in analogous way, we can write $\mathbf{I}_y \otimes \mathbf{W}_R$. Multiplying left and right of matrix $\hat{\mathbf{A}}$ with this two terms ($\mathbf{I}_y \otimes \mathbf{W}_L^H$ and $\mathbf{I}_y \otimes \mathbf{W}_R$, respectively), gives us matrix $\mathbf{B}$:

$$\begin{aligned} \mathbf{B} &= (\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \hat{\mathbf{A}}\ (\mathbf{I}_y \otimes \mathbf{W}_R) \\ &= (\mathbf{I}_y \otimes \mathbf{W}_L^H)\ [\mathbf{I}_y \otimes (\mathbf{A}_x - \mathbf{K}_x) + (\mathbf{A}_y - \mathbf{K}_y) \otimes \mathbf{I}_x]\ (\mathbf{I}_y \otimes \mathbf{W}_R) \\ &= \mathbf{I}_y \otimes \left[\mathbf{W}_L^H(\mathbf{A}_x - \mathbf{K}_x)\mathbf{W}_R\right] + \\ &\quad\ (\mathbf{I}_y \otimes \mathbf{W}_L^H)\,[(\mathbf{A}_y - \mathbf{K}_y) \otimes \mathbf{I}_x]\,(\mathbf{I}_y \otimes \mathbf{W}_R) \\ &= \mathbf{I}_y \otimes \Lambda + \left[\mathbf{I}_y \otimes (\mathbf{W}^H \mathbf{W})\right] [(\mathbf{A}_y - \mathbf{K}_y) \otimes \mathbf{I}_x] \\ &= \mathbf{I}_y \otimes \Lambda + (\mathbf{I}_y \otimes \mathbf{I}_x)((\mathbf{A}_y - \mathbf{K}_y) \otimes \mathbf{I}_x), \end{aligned} \tag{7.44}$$

leading to

$$\mathbf{B} = \mathbf{I}_y \otimes \Lambda + (\mathbf{A}_y - \mathbf{K}_y) \otimes \mathbf{I}_x, \tag{7.45}$$

which is a matrix consisting of three diagonals. In the previous Expressions (7.44) and (7.45), some properties of the Kronecker product, as given in Section

[2] In (7.42), we have assumed that $\mathbf{A}_x - \mathbf{K}_x$ is not defect, i.e., all eigenvalues of $\mathbf{A}_x - \mathbf{K}_x$ have the same algebraic and geometric multiplicity. However, since $\mathbf{A}_x - \mathbf{K}_x$ is complex-symmetric in our case, the latter statement is always satisfied, see Lay [30].

3.1, have been used. Matrix **B** can be expressed as follows:

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{U}_1 & & \\ \mathbf{L}_1 & \mathbf{B}_2 & \ddots & \\ & \ddots & \ddots & \mathbf{U}_{n-1} \\ & & \mathbf{L}_{n-1} & \mathbf{B}_n \end{bmatrix}, \tag{7.46}$$

where all $\mathbf{B}_i, \mathbf{L}_i$ and $\mathbf{U}_i$ are diagonal submatrices.

Permutation Matrix P

Let us define a *permutation matrix* **P** such that the non-zero terms of **P** are

$$\mathbf{P}(m + (n-1)M, n + (m-1)N) = 1, \tag{7.47}$$

for $m = 1, 2, \ldots, M$ and $n = 1, 2, \ldots, N$. Then, by definition, we have $\mathbf{P}^{-1}\mathbf{P} = \mathbf{P}^T\mathbf{P} = \mathbf{I}$ and moreover, **P** is symmetric if $M = N$. [3]

For example, a 9×9 matrix **P** of size has exactly 9 non-zeros at the positions:

$$\begin{array}{lll} \mathbf{P}(1,1) & \mathbf{P}(1+M,2) & \mathbf{P}(1+2M,3) \\ \mathbf{P}(2,1+N) & \mathbf{P}(2+M,2+N) & \mathbf{P}(2+2M,3+N) \\ \mathbf{P}(3,1+2N) & \mathbf{P}(3+M,2+2N) & \mathbf{P}(3+2M,3+2N) \end{array} \tag{7.48}$$

Observe that each row or column of **P** consists of exactly one non-zero element.

Matrix D

Using the permutation matrix **P**, we obtain a *block diagonal matrix* **D** of the form

$$\mathbf{D} = \mathbf{P}^T\mathbf{B}\mathbf{P}, \tag{7.49}$$

which is equivalent to

$$\mathbf{D} = \mathbf{P}\mathbf{B}\mathbf{P}, \tag{7.50}$$

if $M = N$. This is illustrated in Example E.2 of **Appendix E**. Matrix **D**, as defined in (7.49), consists of M blocks $\mathbf{D}_m$ of the form:

$$\begin{bmatrix} \mathbf{D}_1 & & & \\ & \mathbf{D}_2 & & \\ & & \ddots & \\ & & & \mathbf{D}_m \end{bmatrix}, \tag{7.51}$$

where $\mathbf{D}_i$ are submatrices with sizes $N \times N$. Now, each block m is equal to:

$$\mathbf{D}_m = \lambda_m \mathbf{I}_y + \mathbf{A}_y - \mathbf{K}_y, \tag{7.52}$$

which can be easily checked, using (7.45) and (7.49).

[3] since $\mathbf{P}(m+(n-1)M, n+(m-1)N) = \mathbf{P}(m+(n-1)N, n+(m-1)M) = \mathbf{P}^T(n+(m-1)M, m+(n-1)N)$, we obtain $\mathbf{P} = \mathbf{P}^T$ if $M = N$.

7.1.5 Block-Diagonal Linear System $\mathbf{Dv} = \mathbf{g}$

Consider again the linear system

$$\hat{\mathbf{A}}\mathbf{p} = \mathbf{f}, \tag{7.53}$$

with $\hat{\mathbf{A}}$ as in (7.39). Then, this can be written as follows:

$$\begin{aligned} (\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \hat{\mathbf{A}}\ (\mathbf{I}_y \otimes \mathbf{W}_R)[\mathbf{I}_y \otimes \mathbf{W}_R)^{-1}\mathbf{p}] &= (\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{f}; \\ \mathbf{B}\ [(\mathbf{I}_y \otimes \mathbf{W}_R)^{-1}\mathbf{p}] &= (\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{f}; \\ \mathbf{B}\ [(\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{p}] &= (\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{f}. \end{aligned} \tag{7.54}$$

Using permutation matrix $\mathbf{P}$, we can rewrite (7.54) in

$$\begin{aligned} \mathbf{P}^T\ \mathbf{B}\ \mathbf{P}\ [\mathbf{P}^{-1}(\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{p}] &= \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{f}; \\ \mathbf{D}\ [\mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{p}] &= \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{f}. \end{aligned} \tag{7.55}$$

Expressions (7.55) are equivalent to the block diagonal system

$$\mathbf{Dv} = \mathbf{g}, \tag{7.56}$$

with $\mathbf{v} = \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{p}$ and $\mathbf{g} = \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{f}$. As a consequence,

$$\mathbf{p} = (\mathbf{I}_y \otimes \mathbf{W}^R)\mathbf{Pv}. \tag{7.57}$$

In Example E.3, we give an illustration of the transformation of the solution of $\hat{\mathbf{A}}\mathbf{p} = \mathbf{f}$ into the solution of $\mathbf{Dv} = \mathbf{g}$.

Subblocks $\mathbf{D}_m$

Now, we are able to decompose $\mathbf{v}$ and $\mathbf{g}$ in M blocks $\mathbf{v}_m$ and $\mathbf{g}_m$ of size N. The solution can then be obtained by solving the M independent systems

$$\mathbf{D}_m\mathbf{v}_m = \mathbf{g}_m, \tag{7.58}$$

using $\mathbf{D}_m = \lambda_m\mathbf{I}_y + \mathbf{A}_y - \mathbf{K}_y$ of Expression (7.52). These systems have one dimension less than the original system $\hat{\mathbf{A}}\mathbf{p} = \mathbf{f}$. In this way, the solution of the 2-D problem of size MN is obtained by solving M 1-D problems of size N, which is very favorable, especially when M or N is relatively large.

In Example E.4 of **Appendix E** we show how a full block-diagonal system $\mathbf{D}$ can be divided into a few subsystems $\mathbf{D}_m$.

When k is separable and Dirichlet conditions hold at the boundaries, we can always replace the original system $\mathbf{Ap} = \mathbf{f}$ in system (7.56), which is more efficient to solve, since it is separable in linear subsystems. However, in practical applications k is non-separable and also absorbing boundary conditions hold, thus the SoV technique, described in this subsection, is not exact but can be used as a preconditioner, see the next subsection. [4]

[4] In fact, the SoV method is only exact if the wavenumber $k(x, y)$ is constant in Ω. This is the result of the assumption $\tilde{k} = 0$ and the approximations for $\tilde{\gamma}$ in (7.17).

7.1.6 Preconditioned System

The SoV technique as defined in the previous section leads to the SoV preconditioner. The fact that this technique can not be used to decompose the original system $\mathbf{Ap} = \mathbf{f}$ exactly, follows from the following causes:

- the wavenumber $k(x, y)$ has been decomposed into three parts (see expression (7.33)), where the third part (i.e., $\tilde{k}(x, y)$) has been neglected. Then we have obtained $\hat{k}(x, y)$. If k is non-separable, it yields $\hat{k}(x, y) \neq k(x, y)$;
- we have computed averaged values at the boundaries, see the expressions in (7.17). In general, we have $\gamma \neq \tilde{\gamma}$; [5]

Instead of solving the original sytem $\mathbf{Ap} = \mathbf{f}$, we solve the preconditioned system

$$\mathbf{M}_{SoV}^{-1}\mathbf{Ap} = \mathbf{M}_{SoV}^{-1}\mathbf{f} \tag{7.59}$$

where $\mathbf{M}_{SoV}$ is a matrix, which resembles $\mathbf{A}$ after the approximations mentioned above. In fact,

$$\mathbf{M}_{SoV} = \hat{\mathbf{A}}, \tag{7.60}$$

where matrix $\hat{\mathbf{A}}$ is the same as in (7.39).

In the preconditioned iterative methods, as given in **Appendix B**, the linear system $\mathbf{M}_{SoV}\mathbf{x} = \mathbf{b}$ has to be solved. In previous subsections we have shown how to do this efficiently by dividing it in subsystems as seen in (7.58).

7.1.7 Remarks

Note that combining (7.44) and (7.49) leads to

$$\begin{array}{rcl}\mathbf{B} & = & \mathbf{P}^{-T}\mathbf{D}\mathbf{P}^{-1} = \mathbf{PDP}^{T} \\ & = & (\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \mathbf{A}\ (\mathbf{I}_y \otimes \mathbf{W}_R),\end{array} \tag{7.61}$$

using the earlier mentioned property $\mathbf{P}^{-1}\mathbf{P} = \mathbf{P}^T\mathbf{P} = \mathbf{I}$. Now, from (7.61) the approximate version of $\mathbf{A}$, denoted by $\widetilde{\mathbf{A}}$, can be given:

$$\begin{array}{rcl}\hat{\mathbf{A}} & = & (\mathbf{I}_y \otimes \mathbf{W}_L^H)^{-1}\mathbf{PDP}^T(\mathbf{I}_y \otimes \mathbf{W}_R)^{-1} \\ & = & (\mathbf{I}_y \otimes \mathbf{W}_R)\mathbf{PDP}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H).\end{array} \tag{7.62}$$

Now the preconditioner $\mathbf{M}_{SoV} = \widetilde{\mathbf{A}}$ is applied in the preconditioned system (7.59). Moreover, note that the inverse of $\mathbf{M}_{SoV}^{-1}$ can be computed due to

$$\begin{array}{rcl}\mathbf{M}_{SoV}^{-1} = \hat{\mathbf{A}}^{-1} & = & (\mathbf{I}_y \otimes \mathbf{W}_L^H)^{-1}\mathbf{P}^{-T}\mathbf{D}^{-1}\mathbf{P}^{-1}(\mathbf{I}_y \otimes \mathbf{W}_R)^{-1} \\ & = & (\mathbf{I}_y \otimes \mathbf{W}_R)\mathbf{PD}^{-1}\mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H),\end{array} \tag{7.63}$$

where matrix $\mathbf{D}^{-1}$ can be derived by taking $\mathbf{D}_m^{-1}$ for each subblock. However, in practical situations this can be expensive.

[5] Moreover, another cause is the eigenvalues and eigenfunctions (see (7.42)), which are often approximated in (3-dimensional) practical situations, to restrict the computational work and time.

7.1.8 Conclusions in Paper [37]

In Plessix & Mulder [37], the HP is iteratively solved by using Bi-CGSTAB in combination with the SoV preconditioner (7.59). The final conclusions in that article are the following:

- the convergence rate of the iterative methods depends on the frequency and on the roughness of the velocity model;
- when the wavenumber varies only in one dimension (i.e., with one-dimensional models), this preconditioner is efficient;
- for smooth models and low frequencies, the convergence rate is satisfactory;
- for complex models (for instance wedge and Marmousi models, as given in [37]) the method is *not* good enough when the frequency increases;
- it is *not* possible to find a better decomposition of the wavenumber that would improve the convergence rate of the approach, using numerical examples and a mathematical explanation.

7.2 Comparison of the norms of $\mathbf{K}_x, \mathbf{K}_y$ and $\widetilde{\mathbf{K}}$

We have seen that $\widetilde{\mathbf{K}} = 0$ has been assumed in the SoV preconditioner. This preconditioner works very well if $\widetilde{\mathbf{K}}$ is relatively small, i.e., if $||\widetilde{\mathbf{K}}||_2 \ll ||\mathbf{K}_x||_2$ and $||\widetilde{\mathbf{K}}||_2 \ll ||\mathbf{K}_y||_2$. [6] We expect that SoV fails when $\widetilde{\mathbf{K}}$ in norm becomes too large. Therefore, we investigate the norms between these matrices $\mathbf{K}_x, \mathbf{K}_y$ and $\widetilde{\mathbf{K}}$ in this section. In Table 7.1, one can find some results of the comparisons of the norms in our test problems.

In the last column of Table 7.1, we have computed the ratio ρ between $\widetilde{\mathbf{K}}$ and the other matrices using:

$$\rho = \frac{||\widetilde{\mathbf{K}}||_2}{||\mathbf{K}_x||_2 + ||\mathbf{K}_y||_2 + ||\widetilde{\mathbf{K}}||_2} \times 100\%. \tag{7.64}$$

In Table 7.1, one may observe the relation between the number of required iterations of using SoV and the value of the ratio ρ. Indeed, if ρ becomes smaller and, therefore, the relative contribution of $\widetilde{\mathbf{K}}$ becomes smaller, the iterative method with SoV requires less iterations.

To enhance the SoV preconditioner, we work out the following ideas in the next section:

- In each problem, the quantities $k_x(x)$ and $k_y(y)$ are uniquely determined by construction of (7.2), see also [37]. However, we investigate other choices $k_x(x)$ and $k_y(y)$, such that it may *not* satisfy the integrals in (7.2)

[6] The standard norm of a matrix $\mathbf{A}$, i.e., $||\mathbf{A}||_2$, returns the largest singular value of $\mathbf{A}$. More information about the singular value (decomposition) can be found in e.g. Nakos & Joyner [34].

Test Problem	M, N	Iter.	$\|\|\mathbf{K}\|\|_2$	$\|\|\mathbf{K}_x\|\|_2$	$\|\|\mathbf{K}_y\|\|_2$	$\|\|\widetilde{\mathbf{K}}\|\|_2$	ρ
(R)	15	6	4.62	1.17	0.15	0.07	5.0 %
	25	6	7.71	1.52	0.20	0.07	3.7 %
	35	6	10.79	1.80	0.24	0.07	3.4 %
	45	7	13.87	2.04	0.27	0.07	3.1 %
(V)	15	78	13.4	3.01	1.38	0.62	12.4 %
	25	35	22.4	3.95	1.78	0.59	9.4 %
	35	32	31.3	4.69	2.09	0.63	8.5 %
	45	34	40.3	5.34	2.38	0.65	7.8 %

***Table 7.1:** Comparison of the norms ($\times 10^3$) between the matrices obtained by the decomposition of the wavenumber using test problems (R) and (V) in combination with the wedge model.*

anymore, but which may lead to a smaller $||\widetilde{\mathbf{K}}||_2$ and hence, to a faster convergence;

- another way is to search for a block-diagonal $\widetilde{\mathbf{K}} \neq 0$, denoted by $\widetilde{\mathbf{K}}_{block}$, such that the norm of the difference $||\widetilde{\mathbf{K}}_{block} - \widetilde{\mathbf{K}}_{orig}||_2$ is minimal, where we have denoted the original $\widetilde{\mathbf{K}}$ with $\widetilde{\mathbf{K}}_{orig}$. This block-diagonal structure is required to ensure the system $\mathbf{Dv} = \mathbf{g}$ to be block-diagonal, such that it can still be solved with linear subsystems $\mathbf{D}_m\mathbf{v}_m = \mathbf{g}_m$. Plessix & Mulder [37] have claimed that this $\widetilde{\widetilde{\mathbf{K}}} \neq 0$ is difficult to find, but nevertheless we will investigate this item further.

7.3 Enhancing SoV Preconditioner

In this section, we work out the two ideas mentioned in the previous section.

7.3.1 Alternative Choices for $k_x(x)$ and $k_y(y)$ in SoV

We investigate several alternatives for $k_x(x)$ and $k_y(y)$ in the SoV preconditioner.

In the standard SoV method, we have applied the construction of the wavenumber $k(x, y)$, as seen in (7.3), where $x_a = y_a = 0$ and $x_b = y_b = 1$ are taken. This construction satisfies

$$\begin{cases} k^2(x,y) = k_x^2(x) + k_y^2(y) + \tilde{k}^2(x,y), \\ \int_0^1 \tilde{k^2}(x,y)\,\mathrm{d}x = 0, \quad \forall y, \\ \int_0^1 \tilde{k^2}(x,y)\,\mathrm{d}y = 0, \quad \forall x, \end{cases} \tag{7.65}$$

see also (7.2).

M, N	Standard	(C1)	(C2)
25	35	34	41
35	32	30	36
45	34	34	38

Table 7.2: Number of iterations of Bi-CGSTAB with original SoV (second column), with Choice 1 (third column) and with Choice 2 (fourth column) in test problem (V) in combination with the wedge model.

Consider now the following two alternative choices of $k_x(x)$ and $k_y(y)$:

$$(\mathrm{C1}) = \begin{cases} k_x^2(x) &= \frac{1}{Y^2}\left(\int_0^1 k(x,y)\,\mathrm{d}y\right)^2, \\ k_y^2(y) &= \frac{1}{X}\int_0^1 k^2(x,y) - k_x^2(x)\,\mathrm{d}x, \\ \tilde{k}^2(x,y) &= k^2(x,y) - k_x^2(x) - k_y^2(y), \end{cases} \tag{7.66}$$

and

$$(\mathrm{C2}) = \begin{cases} k_x^2(x) &= 0, \\ k_y^2(y) &= \frac{1}{X}\int_0^1 k^2(x,y)\,\mathrm{d}x, \\ \tilde{k}^2(x,y) &= k^2(x,y) - k_x^2(x) - k_y^2(y). \end{cases} \tag{7.67}$$

In the first expression of (7.3), we have squared the integrand, whereas the whole integral is squared in the first expression of (C1). In (C2), we have taken the integral in the first expression to be zero, resulting in a SoV wavenumber which only depends on the y-direction. Note that the second and third expression of all (7.3), (7.66) and (7.67) are the same. Morever, note also that, in general, these constructions (C1) and (C2) do *not* satisfy the second and third equations of (7.65) anymore.

The corresponding subplots of the original and the alternative SoV wavenumbers in the domain can be found in Figures 7.1.

Some results of the test runs using (C1) and (C2) can be found in Table 7.2. In this table, we observe the slightly better results of (C1) and the somewhat disappointed results of (C2). In general, the choices we have made do not lead to impressive acceleration of the convergence. Hence, we pay no more attention to this aspect in further research.

Next, the spectra of preconditioned systems in test problems using both (C1) and (C2), which are applied in Table 7.2, can be found in **Appendix G**.

7.3.2 Improved $\widetilde{\mathbf{K}}$ in SoV

Recall that

$$\hat{\mathbf{A}} = \mathbf{I}_y \otimes (\mathbf{A}_x - \mathbf{K}_x) + (\mathbf{A}_y - \mathbf{K}_y) \otimes \mathbf{I}_x, \tag{7.68}$$

as in (7.41). However, if we do not assume $\tilde{k}(x, y) = 0$, then we can replace this expression by

$$\mathbf{A} = \mathbf{I}_y \otimes (\mathbf{A}_x - \mathbf{K}_x) + (\mathbf{A}_y - \mathbf{K}_y) \otimes \mathbf{I}_x - \widetilde{\mathbf{K}}. \tag{7.69}$$

Now, matrix **B** in (7.44) turns out to be

$$\mathbf{B} = \mathbf{I}_y \otimes \Lambda + (\mathbf{A}_y - \mathbf{K}_y) \otimes \mathbf{I}_x - (\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \widetilde{\mathbf{K}}\ (\mathbf{I}_y \otimes \mathbf{W}_R). \tag{7.70}$$

Then, instead of (7.49), we obtain

$$\mathbf{D} = \mathbf{P}^T \mathbf{B} \mathbf{P} + \widetilde{\widetilde{\mathbf{K}}}, \tag{7.71}$$

where

$$\widetilde{\widetilde{\mathbf{K}}} = \mathbf{P}^T \widetilde{\mathbf{K}}' \mathbf{P}, \tag{7.72}$$

with $\widetilde{\mathbf{K}}'$ defined by

$$\widetilde{\mathbf{K}}' = (\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \widetilde{\mathbf{K}}\ (\mathbf{I}_y \otimes \mathbf{W}_R). \tag{7.73}$$

Note that, in general, the matrix $\widetilde{\widetilde{\mathbf{K}}}$ has no diagonal or block-diagonal structure, see also Example E.5 in **Appendix E**.

In this subsection we investigate possibilities for a *block-diagonal* matrix $\widetilde{\widetilde{\mathbf{K}}} \neq \mathbf{0}$ in such a way that this leads to better convergence results for the HP.

Diagonal $\widetilde{\widetilde{\mathbf{K}}}$

First, we investigate a *diagonal* $\widetilde{\widetilde{\mathbf{K}}}$. The most simple choice in this case is

$$\widetilde{\widetilde{\mathbf{K}}}_{diag} = \mathrm{diag}(\widetilde{\widetilde{\mathbf{K}}}). \tag{7.74}$$

Due to the structure of $\widetilde{\widetilde{\mathbf{K}}}$, we obtain exactly the same matrix if we take

$$\widetilde{\widetilde{\mathbf{K}}}_{block} = \mathrm{block}(\widetilde{\widetilde{\mathbf{K}}}), \tag{7.75}$$

where block($\widetilde{\widetilde{\mathbf{K}}}$) denotes the diagonal blocks of $\widetilde{\widetilde{\mathbf{K}}}$. Thus $\widetilde{\widetilde{\mathbf{K}}}_{block} = \widetilde{\widetilde{\mathbf{K}}}_{diag}$. We illustrate this with Example E.6, see **Appendix E**.

Next, we obtain

$$\widetilde{\mathbf{K}}_{mod} = (\mathbf{I}_y \otimes \mathbf{W}_R)\mathbf{P}\ \widetilde{\widetilde{\mathbf{K}}}_{diag}\ \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H). \tag{7.76}$$

Now, we replace $\widetilde{\mathbf{K}} = 0$ by $\widetilde{\mathbf{K}} = \widetilde{\mathbf{K}}_{mod}$ in the SoV preconditioner .

Results & Discussion

Before we give the results of our test runs using $\widetilde{\mathbf{K}}_{mod}$, we note that if we know $\mathbf{W}_R$, then $\mathbf{W}_L^H$ can be computed / approximated in two ways:

1. $\widetilde{\mathbf{W}}_L^H = \mathbf{W}_R^H$;
2. $\mathbf{W}_L^H = \mathbf{W}_R^T$.

These two choices are used in the test runs.

The first choice is the *naive* approach. $\widetilde{\mathbf{W}}_L^H$ is only exact if $\mathbf{A}$ is Hermitian, see Theorem 3.4. However, our matrix $\mathbf{A}$ is not Hermitian due to the imaginary components. Therefore, $\widetilde{\mathbf{W}}_L^H$ is only an *approximation* of $\mathbf{W}_L^H$.

The *exact* approach is to choose $\mathbf{W}_L^H = \mathbf{W}_R^{-1}$, see Theorem 3.3. $\mathbf{W}_R^{-1}$ is expensive to compute in practice. Fortunately, we deal with a *complex-symmetric* $\mathbf{A}$ and, hence, we can apply Theorem 3.5. We obtain $\overline{\mathbf{W}}_R = \mathbf{W}_L$, where $\overline{\mathbf{W}}_R$ denotes the *conjugate* of $\mathbf{W}_R$. This leads to the second choice $\mathbf{W}_L^H = \mathbf{W}_R^T$.

We denote $\widetilde{\mathbf{K}}_{mod}$ obtained with the first and second choice, by $\widetilde{\mathbf{K}}_{mod(1)}$ and $\widetilde{\mathbf{K}}_{mod(2)}$, respectively. The new SoV preconditioners are in these cases:

$$\mathbf{M}_{mod(i)} = \mathbf{M}_{SoV} + \widetilde{\mathbf{K}}_{mod(i)}, \quad i = 1, 2. \tag{7.77}$$

where $\mathbf{M}_{SoV}$ is the original SoV preconditioner with $\widetilde{\mathbf{K}} = 0$. Note further that the computational work to obtain $\widetilde{\mathbf{K}}_{mod(i)}$ is relatively cheap compared to solving $\mathbf{M}_{mod(i)}\mathbf{x} = \mathbf{b}$.

In Table 7.3, one can find the results applying this new SoV preconditioner.

M, N	$\mathbf{M}_{SoV}$	$\mathbf{M}_{mod(1)}$	$\mathbf{M}_{mod(2)}$
15	78	68	58
25	35	35	35
35	33	29	30
45	35	33	33

Table 7.3: Number of iterations of Bi-CGSTAB using M_{SoV}, $M_{mod(1)}$ and $M_{mod(2)}$, respectively. Test problem (V) is applied with the wedge model.

In Table 7.3, we can see the decrease of the number of iterations using $\mathbf{M}_{mod(i)}$ compared to $\mathbf{M}_{SoV}$. However, the differences between $\mathbf{M}_{SoV}$ and both $\mathbf{M}_{mod(i)}$ are relatively small, especially for sufficiently large M and N which is of our interest.

Moreover, one observes in Table 7.3 that the methods applying $\widetilde{\mathbf{K}}_{mod(1)}$ and $\widetilde{\mathbf{K}}_{mod(2)}$ give approximately the *same* number of iterations, which is rather remarkable, see **Appendix H**.

In further analysis, we apply only $\widetilde{\mathbf{K}} = \widetilde{\mathbf{K}}_{mod(2)}$, which is denoted by $\widetilde{\mathbf{K}}_{mod}$ for simplicity.

Structure of $\widetilde{\mathbf{K}}'$ and $\widetilde{\widetilde{\mathbf{K}}}$

A reason of the small advantage of using $\widetilde{\mathbf{K}} = \widetilde{\mathbf{K}}_{mod}$, which can be seen Table 7.3, becomes clear when we look at the structure of $\widetilde{\mathbf{K}}'$ and $\widetilde{\widetilde{\mathbf{K}}}$. The plots of $\widetilde{\mathbf{K}}'$ and $\widetilde{\widetilde{\mathbf{K}}}$ of our problem with different gridsizes ($M, N = 5, 15, 25$) can be found in Figures 7.2–7.4.

Considering Figures 7.2–7.4, one can make the following observations:

Observation 1 increasing the number of elements M and N, leads to a *block-diagonal* matrix $\widetilde{\mathbf{K}}'$ which has relatively small diagonal elements;

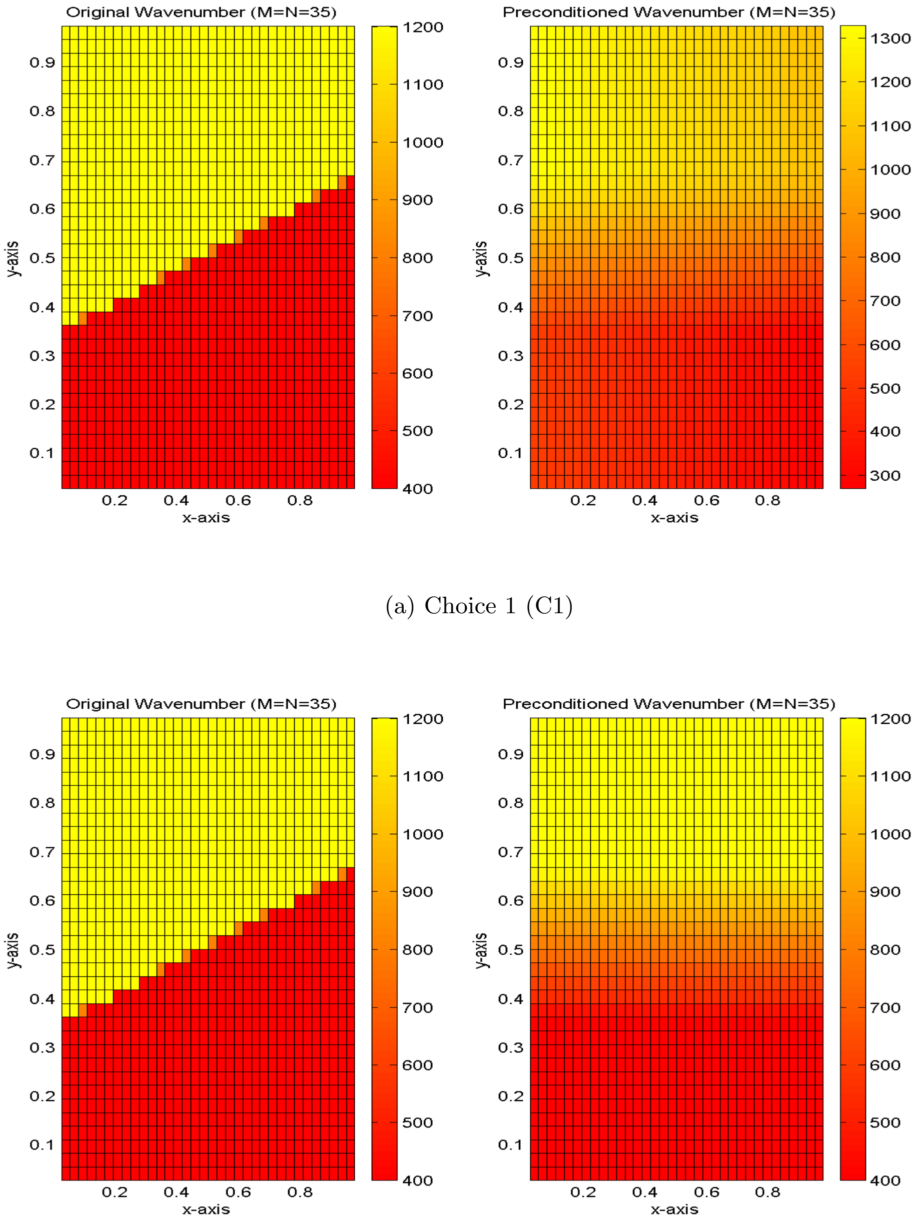
Observation 2 however, considering the *full* matrix $\widetilde{\widetilde{\mathbf{K}}}$ the diagonal elements are obviously significant.

Note that, in fact, Observation 2 is in contradictory to the observation made in Plessix & Mulder [37], since they have concluded that $\widetilde{\mathbf{K}}'$ is not diagonal dominant and moreover, the terms on the diagonal are almost zero. Apparently, our test problems are too small to find comparable results with those of [37].

7.3.3 Ideas for Future Research

We have investigated various choices of $k_x(x)$ and $k_y(y)$, but the differences between them were relatively small.

Examining modified forms of $\widetilde{\mathbf{K}}$, instead of taking $\mathbf{K} = \mathbf{0}$ in the original SoV preconditioner, leads to the following first result: taking $\widetilde{\mathbf{K}}_{mod} = \widetilde{\mathbf{K}}_{diag}$ gives a somewhat better convergence. More research in $\widetilde{\mathbf{K}}_{mod}$ is needed to improve the convergence behaviour. In **Appendix I**, we show that it should be possible to find such a $\widetilde{\mathbf{K}}_{mod}$. We have done some small experiments using so-called *mass lumping* techniques (see e.g. Van Kan & Segal [26]), which do not make sense, since they do not improve the convergence rate of iterative methods. Mass lumping of a matrix means that for each row all non-diagonal elements are summed and added to the diagonal element in a 'sensible' way.



(a) Choice 1 (C1)

(b) Choice 2 (C2)

Figure 7.1: Wavenumber k for Choice 1 (C1) and Choice 2 (C2). Left subplots: original wavenumber. Right subplots: wavenumber in the preconditioned case (i.e., SoV wavenumber). Gridsizes $M \times N = 35 \times 35$ are applied in these subplots.

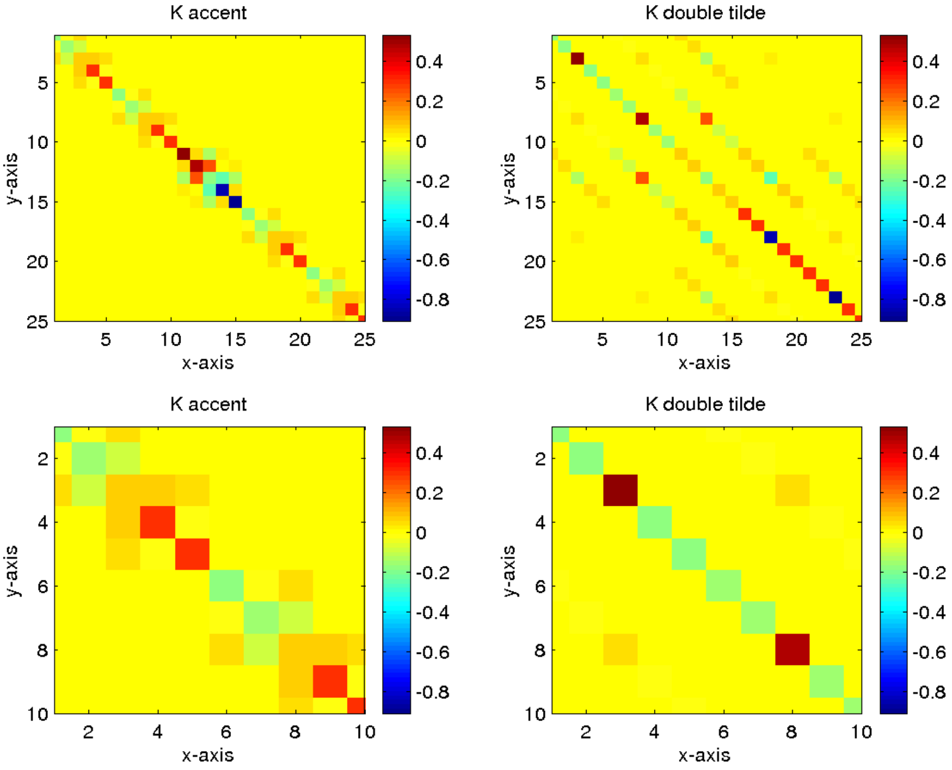


Figure 7.2: On the top: matrices $\widetilde{K}'$ *(left) and* $\widetilde{\widetilde{K}}$ *(right) with* $M, N = 5$*. On the bottom: the enlargements of the plots above. Moreover, the axes are labelled by the number of elements (*$= [1, 2, \ldots, MN]$*).*

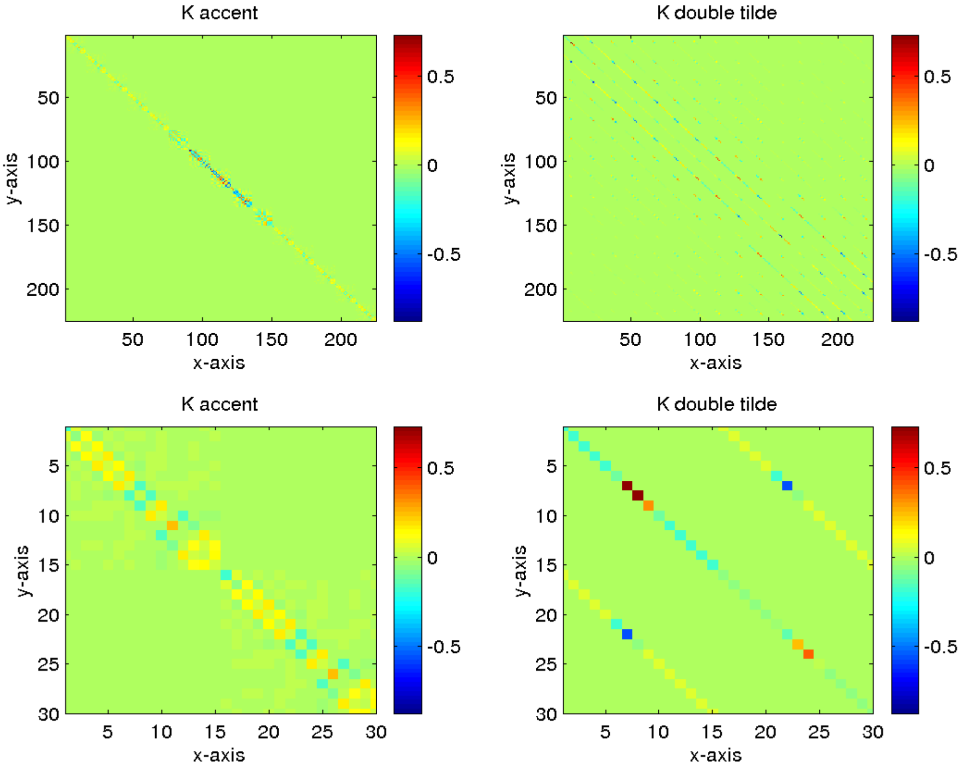


Figure 7.3: On the top: matrices $\widetilde{K}'$ (left) and $\widetilde{\widetilde{K}}$ (right) with $M, N = 15$. On the bottom: the enlargements of the plots above. Moreover, the axes are labelled by the number of elements ($= [1, 2, \ldots, MN]$).

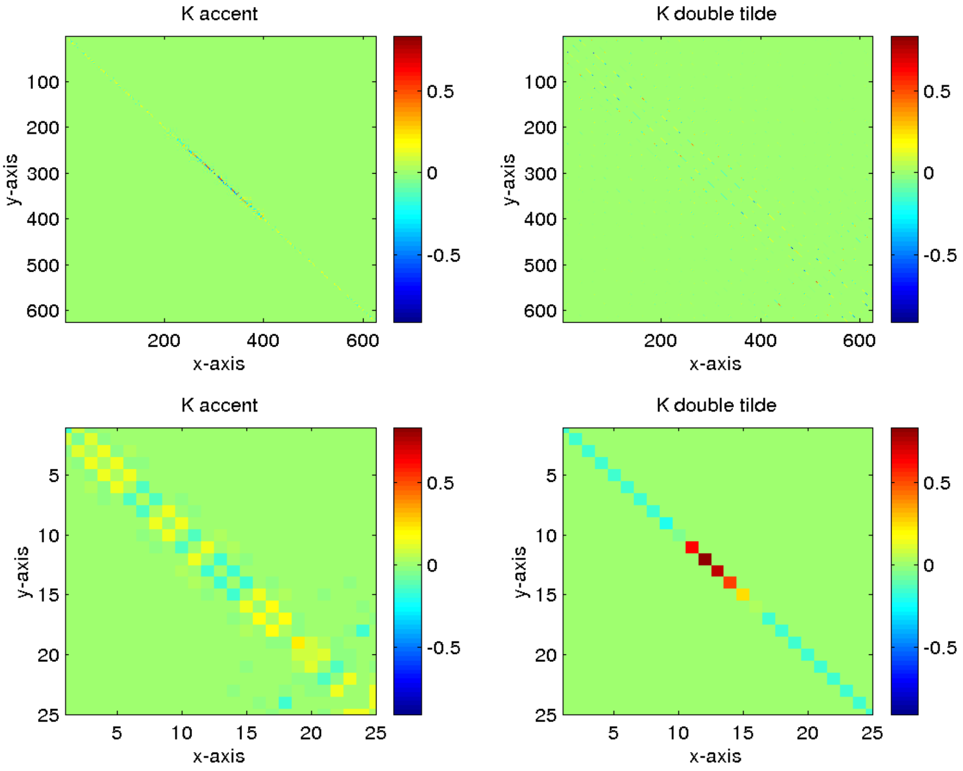


Figure 7.4: On the top: matrices $\widetilde{K}'$ (left) and $\widetilde{\widetilde{K}}$ (right) with $M, N = 25$. On the bottom: the enlargements of the plots above. Moreover, the axes are labelled by the number of elements ($= [1, 2, \ldots, MN]$).

Chapter 8

Eigenvalue and Eigenvector Analysis

As a preparation for the next chapter, where some combined preconditioners will be introduced, we investigate the possibilities for combining the SoV and the CSL preconditioners by using eigenvalue and eigenvector analysis.

We start with analyzing the smallest eigenvalues and their corresponding eigenvectors of the systems $\mathbf{A}$, $\mathbf{M}_{CSL}^{-1}\mathbf{A}$ and $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ in Section 8.1. The aim of this analysis is to examine whether the 'bad' eigenvalues of the SoV preconditioned system correspond with the 'bad' eigenvalues of the original system, where 'bad' eigenvalues means small eigenvalues in absolute sense. As mentioned in Chapter 5, we know that the bad eigenvalues of both the CSL preconditioned and the original systems are related to each other. If we can show that the 'bad' eigenvalues of the SoV preconditioned system are corresponding to *other* eigenvalues of the original system, then a combined preconditioned can make sense.

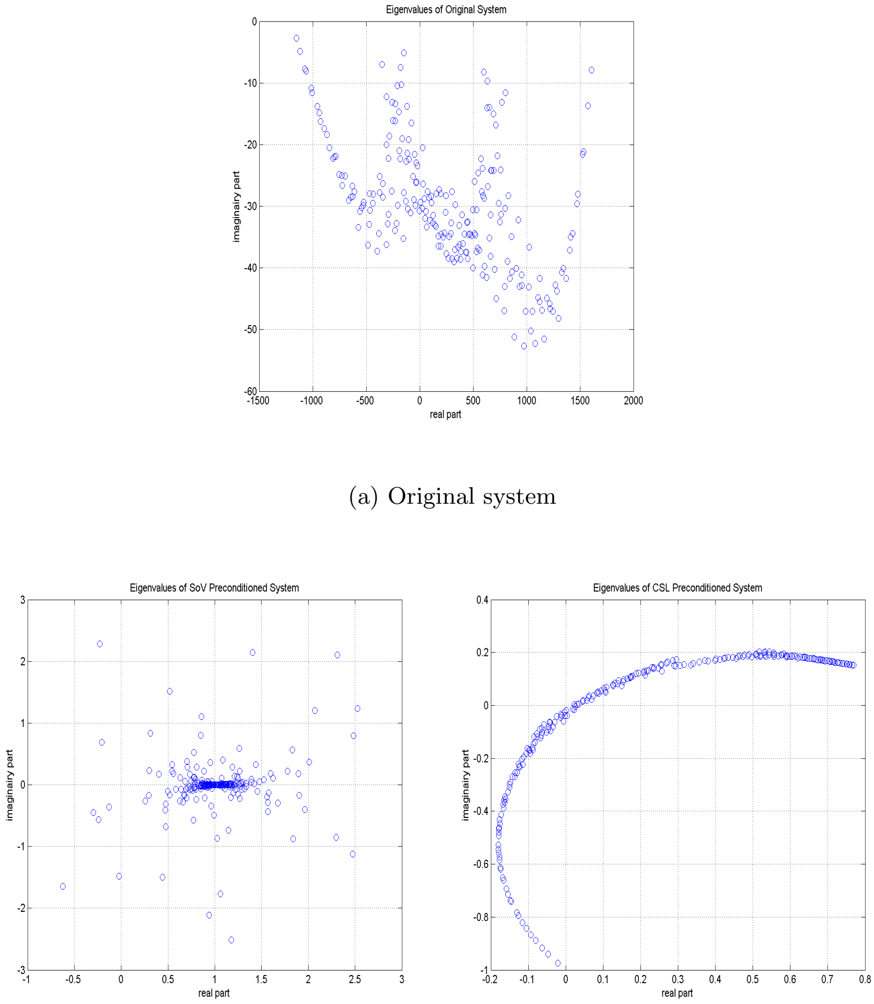
In Sections 8.2 and 8.3, we show the importance of the 'bad' eigenvalues for the solution by considering their eigenvectors. In fact, it is shown that a successful combined preconditioner has to get rid of the these 'bad' eigenvalues, since the corresponding eigenvectors of them determine mainly the solution.

8.1 Smallest Eigenvalues and their Eigenvectors

In this section, we consider the *smallest absolute* values of the (complex) *eigenvalues* and their corresponding *eigenvectors* for the systems $\mathbf{A}$, $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ and $\mathbf{M}_{CSL}^{-1}\mathbf{A}$. We have written the MATLAB program `mineig(X,p)` which computes the p smallest eigenvalues in absolute sense and their eigenvectors for an arbitrary matrix $\mathbf{X}$.

8.1.1 Example 1

In the subplots of Figures 8.2, one can find the results for the case of $p = 4$ and $M, N = 15$ in test problem (V) in combination with the wedge model. Therefore, the eigenvectors in *absolute* sense corresponding to the four smallest eigenvalues (also in absolute sense) are plotted in these figures. The eigenvector belonging to the smallest eigenvalue is plotted on the top, the eigenvector belonging to the second smallest eigenvalue is plotted in the second subplot and so on. Since we consider the 2-D HP we have also 2-D eigenvectors, but they are represented as vectors in 1-D (analogous to Expression (2.19)) in all figures, for simplicity.



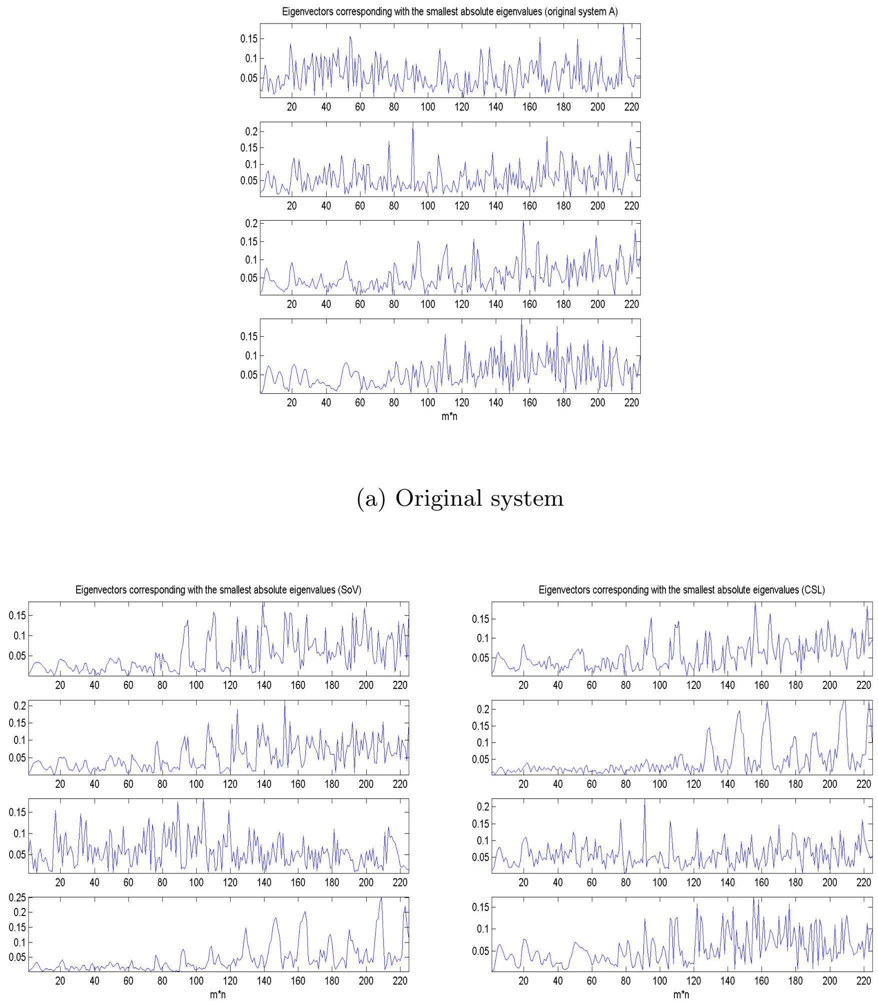
(a) Original system

(b) SoV preconditioned system

(c) CSL preconditioned system

Figure 8.1: Eigenvalues of the systems A, $M_{SoV}^{-1}A$, $M_{CSL}^{-1}A$ with $M, N = 15$ in test problem (V) with the wedge model.

In Figure 8.1, one can observe that it is rather difficult to analyze the four smallest eigenvalues with their eigenvectors, since there are relative many eigenvalues near zero in the original system **A**. If one does try to investigate this and, therefore, to compare the three subplots of Figures 8.2, the following results can be found. There are *no* common eigenvectors for the original and



(a) Original system

(b) SoV preconditioned system

(c) CSL preconditioned system

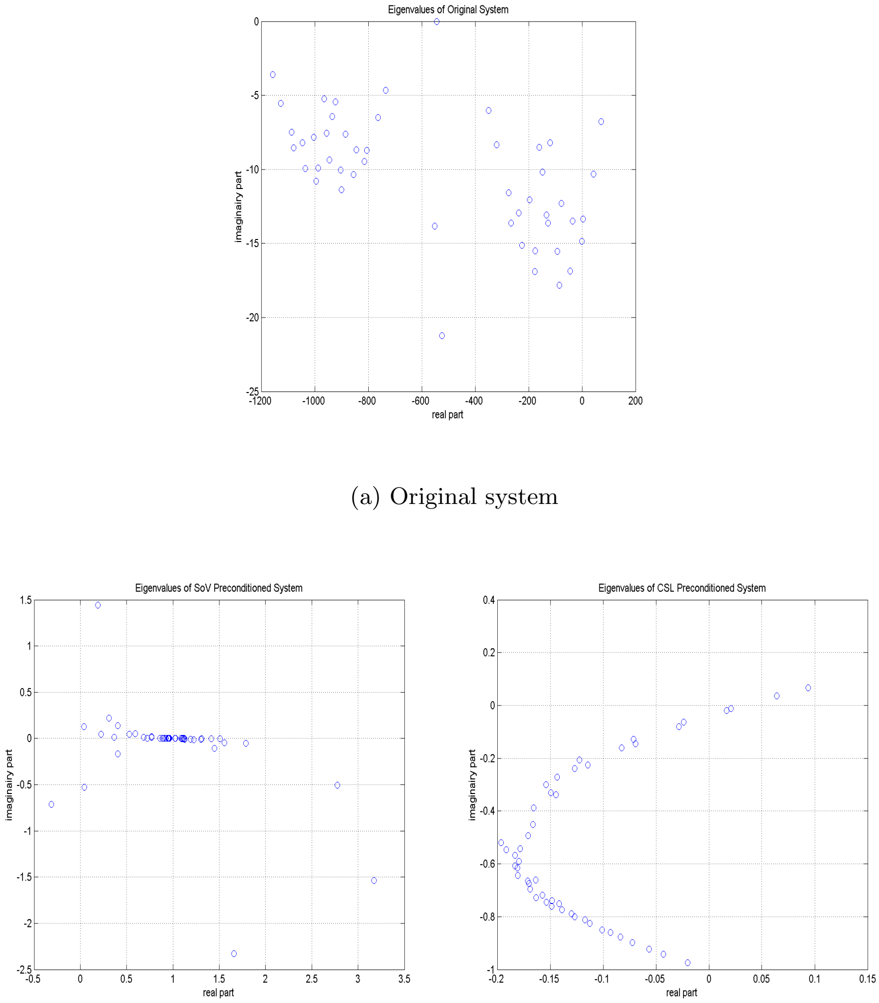
Figure 8.2: Eigenvectors corresponding to the four smallest absolute eigenvalues with $M, N = 15$ ***in test problem (V) with the wedge model.***

SoV preconditioned system, while the original and CSL-preconditioned system show a few common eigenvectors. For example, the *fourth* eigenvector of both Figure 8.2(a) and 8.2(c) seem to be approximately the same. Moreover, the *third* eigenvector of Figure 8.2(a) looks to be equal to the *first* eigenvector of Figure 8.2(c).

8.1.2 Example 2

We take the same example as in the previous subsection, but now with a coarser grid ($M, N = 7$). In this case, there are *fewer* eigenvalues around zeros, which may lead to a *easier* analysis of the smallest eigenvalues and their eigenvectors.

Analogous to Figure 8.1, the eigenvalues are given in Figure 8.3. Thereafter, the results of the eigenvectors can be found in the subplots of Figure 8.4.



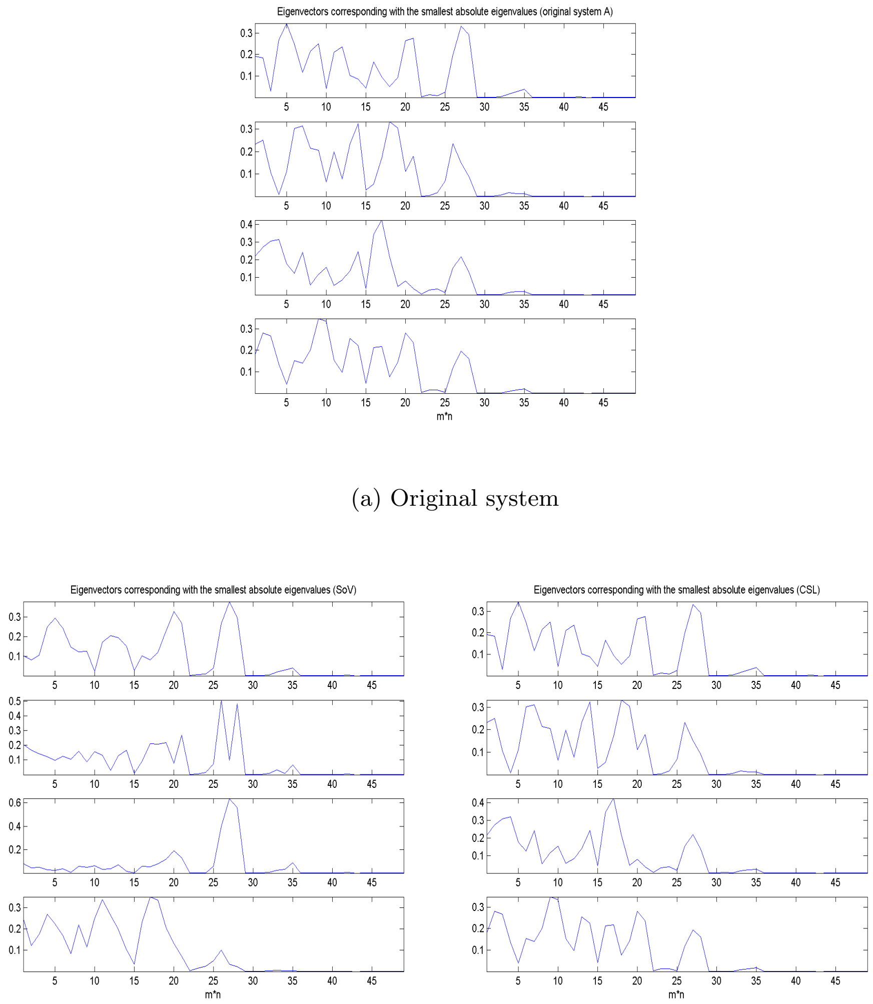
(a) Original system

(b) SoV preconditioned system

(c) CSL preconditioned system

Figure 8.3: Eigenvalues of the systems $\boldsymbol{A}$, $\boldsymbol{M}_{SoV}^{-1}\boldsymbol{A}$, $\boldsymbol{M}_{CSL}^{-1}\boldsymbol{A}$ with $M, N = 7$ in test problem (V) with the wedge model.

We can see again that the original and SoV preconditioned system do not have common eigenvectors, while the original and CSL *do* have these common eigenvectors, which is more clear than in the previous example. All four eigenvectors of Figure 8.4(c) are approximately identical to those of Figure 8.4(a) and even in the same order!



(a) Original system

(b) SoV preconditioned system

(c) CSL preconditioned system

Figure 8.4: Eigenvectors corresponding to the four smallest absolute eigenvalues with $M, N = 7$ in test problem (V) with the wedge model.

8.1.3 Conclusion

In problems with large M and N, it is difficult to analyze the smallest eigenvalues and their eigenvectors. Considering the eigenvectors in examples with relative small M and N, we conclude that indeed the bad eigenvalues of $\mathbf{A}$ are related to the bad eigenvalues of the CSL preconditioned system. Furthermore, there seems to be no relation between the bad eigenvalues of $\mathbf{A}$ and those of the SoV preconditioned system.

There are possibilities with prospects for finding a combination of both preconditioners treating the HP, because the ‘bad’ eigenvalues of $\mathbf{A}$ are also the bad eigenvalues of the CSL preconditioned system, whereas they are *not* the bad eigenvalues of the SoV preconditioned system. We know that CSL gets rid of the relatively large eigenvalues and, hopefully, SoV will get rid of the smaller eigenvalues of matrix **A**. Only in this case, a combined preconditioner will succeed.

8.2 Approximated Solution using Eigenvectors

In Theorem 3.7, we have proved that solution $\mathbf{p}$ of the linear system $\mathbf{Ap} = \mathbf{f}$, where $\mathbf{A}$ is a complex-symmetric matrix with distinct eigenvalues, can be written as the linear combination of the eigenvectors $\mathbf{v}_i$, i.e.,

$$\mathbf{p} = c_1\mathbf{v}_1 + \ldots + c_n\mathbf{v}_n, \tag{8.1}$$

where the coefficients c_i are equal to

$$c_i = \frac{\langle \mathbf{p}, \mathbf{v}_i \rangle}{\langle \mathbf{v}_i, \mathbf{v}_i \rangle}, \tag{8.2}$$

assuming that $\langle \mathbf{v}_i, \mathbf{v}_i \rangle \neq 0 \; \forall i = 1, \ldots, n$ and taking the conjugate inner product as our standard inner product.

The consequence of this theorem is that if $k < n$ then $\mathbf{p}_k$ defined by

$$\mathbf{p}_k = c_1\mathbf{v}_1 + \ldots + c_k\mathbf{v}_k, \tag{8.3}$$

is an approximation of $\mathbf{p}$. Note that if $k = n$ then $\mathbf{p} = \mathbf{p}_n$.

Now we turn back to Examples 1 and 2 of the previous section and consider again the figures drawn in these examples, where the eigenvectors corresponding to the four smallest eigenvalues of the various preconditioned systems have been depicted. In the extensions of the examples, we compare the vector $\mathbf{p}_4$ with $\mathbf{p}$ for both the systems $\mathbf{M}_{CSL}^{-1}\mathbf{A}$ and $\mathbf{M}_{SoV}^{-1}\mathbf{A}$, as can be seen in the next subsections.

We can apply Theorem 3.7 for the complex-symmetric matrix $\mathbf{A}$ (where we assume the eigenvalues to be distinct). However, it is *not* applicable to matrices $\mathbf{M}_{CSL}^{-1}\mathbf{A}$ and $\mathbf{M}_{SoV}^{-1}\mathbf{A}$, since these are, in general, *not* complex-symmetric, i.e., the corresponding eigenvectors are *not* orthogonal, generally! In the case of $\mathbf{M}_{CSL}^{-1}\mathbf{A}$, the coefficients of Theorem 3.7 can still be computed if M and N are sufficient small, because the 'bad' eigenvectors are the same for $\mathbf{M}_{CSL}^{-1}\mathbf{A}$ and $\mathbf{A}$, see e.g. Figure 8.4. For relatively large M and N, it is useless to compute coefficients c_i for matrices $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ and $\mathbf{M}_{CSL}^{-1}\mathbf{A}$, as given in Theorem 3.7. It gives at most a poor approximation of the real $\mathbf{p}_4$.

8.2.1 Extension of Example 1

The results of the comparison of $\mathbf{p}_4$ and $\mathbf{p}$ for the original and the preconditioned systems can be found in Figure 8.5. The approximation of $\mathbf{p}$, using the eigenvectors corresponding to the smallest eigenvalues of $\mathbf{A}$, is given with a straight narrow line (approximated solution $\mathbf{A}$).

Note first that the approximated solution corresponding to $\mathbf{A}$ differs from those of $\mathbf{M}_{CSL}^{-1}\mathbf{A}$ and $\mathbf{M}_{SoV}^{-1}\mathbf{A}$. Although it seems that the eigenvectors of $\mathbf{M}_{CSL}^{-1}\mathbf{A}$ resemble the numerical solution well, it can be observed that the local maxima of the plots are situated at different locations. Hence, the results of $\mathbf{M}_{CSL}^{-1}\mathbf{A}$ and $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ give a bad approximation of the solution and, therefore, they are indeed useless.

Since there are in total $15 \times 15 = 225$ eigenvectors, it is obvious that only four eigenvectors (even corresponding to the smallest eigenvalues) are not sufficient

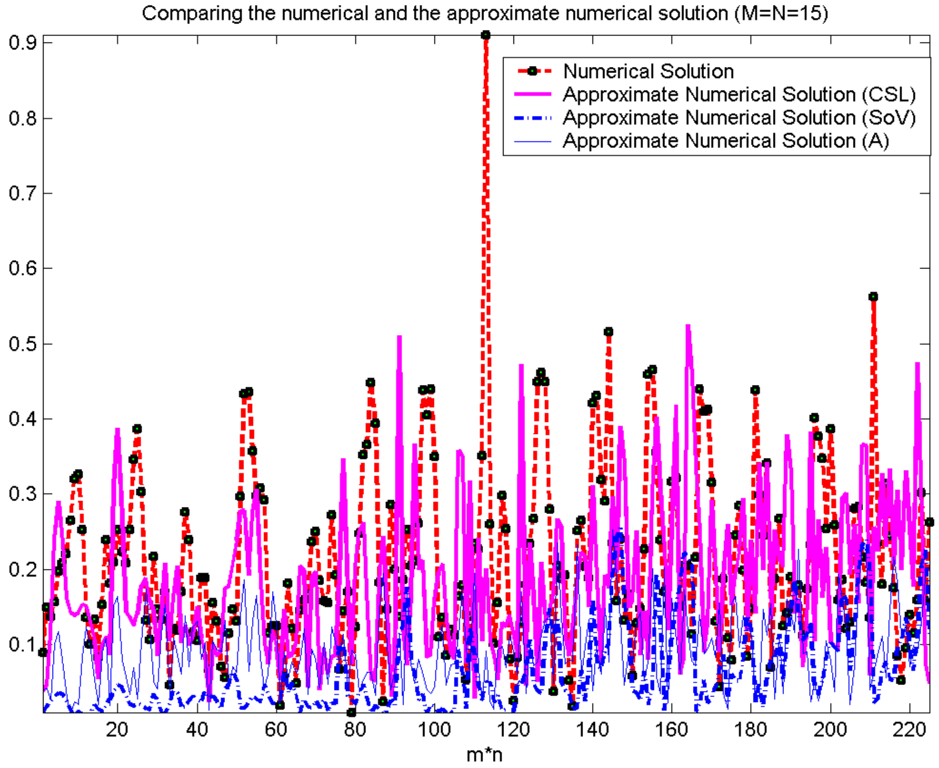


Figure 8.5: Approximated solution using the eigenvectors corresponding to the four smallest absolute eigenvalues of $M_{CSL}^{-1}A$ in the case of $M, N = 15$.

to represent the solution well. Moreover, there are a lot of eigenvalues around zero in the original system **A** (see also Figure 8.1(a)), whereas we have only taken four eigenvalues to approximate the solution. The expectation is that, in the case of $M, N = 7$, the results will be better, see the next subsection.

If we take 20 (of the possible 225) instead of four 'bad' eigenvectors, then one can find the results in Figure 8.6. In this case, we observe that the 20 eigenvectors resemble the numerical solution rather well.

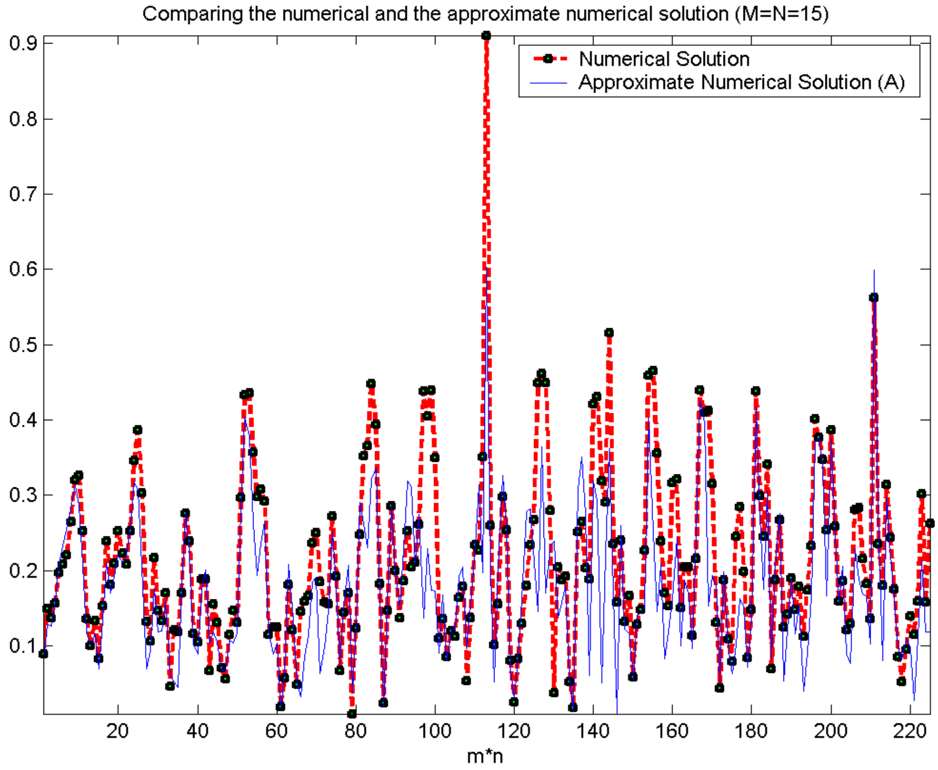


Figure 8.6: Approximated solution using the eigenvectors corresponding to 20 smallest absolute eigenvalues of A in the case of $M, N = 15$*.*

8.2.2 Extension of Example 2

The results, analogous to Example 1 of the previous subsection, can be found in Figure 8.7.

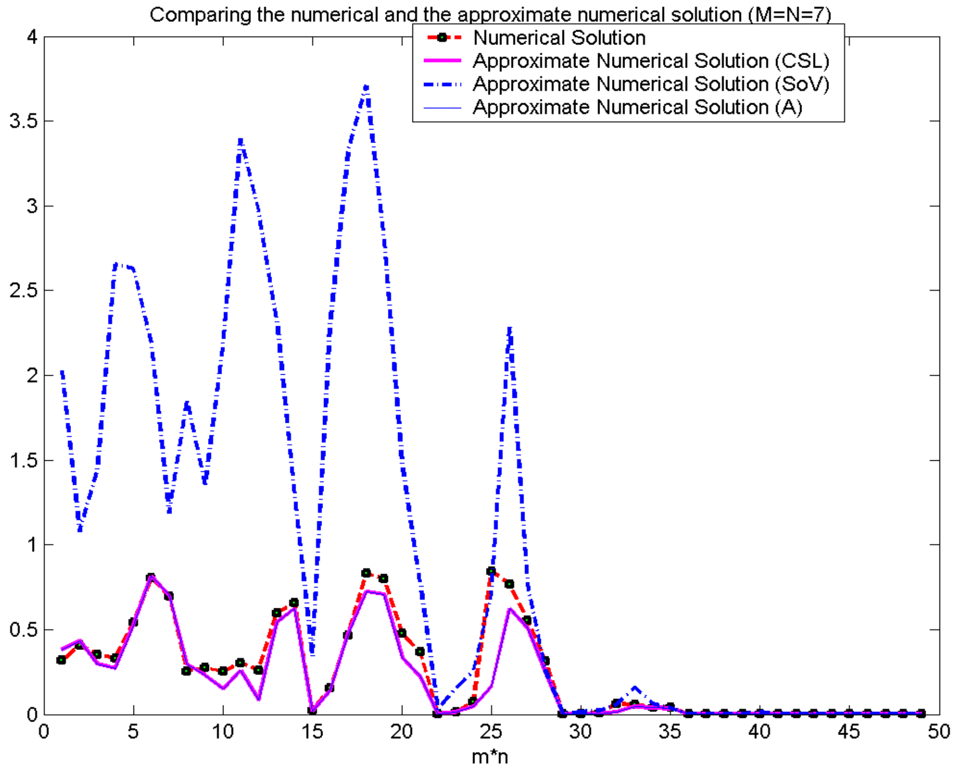


Figure 8.7: Approximated solution using the eigenvectors corresponding to the four smallest absolute eigenvalues of matrices $M^{-1}A$ and A in the case of $M, N = 7$.

Note that the approximation of solution $\mathbf{p}$ based on the bad eigenvectors of the SoV preconditioned system is poor, while the approximation based on the four (of a total of $7 \times 7 = 49$) eigenvectors of the SoV preconditioned and the original system are the same! This can also be seen in Figure 8.8, which is almost the same as Figure 8.7 and only the case $\mathbf{M}_{SoV}^{-1}\mathbf{A}$ is omitted to make the other plots more clear.

8.2.3 Conclusion

Taking about 10% of the number of the bad eigenvectors of $\mathbf{A}$ gives a good approximation of the numerical solution. This holds also for CSL preconditioned system $\mathbf{M}_{CSL}^{-1}\mathbf{A}$, for sufficiently small M and N.

Therefore, in future research of combining preconditioners, one has to look for a variant which is able to deal with the 'bad' eigenvalues and eigenvectors. A succesfully combined preconditioner should be robust for these bad eigenvalues.

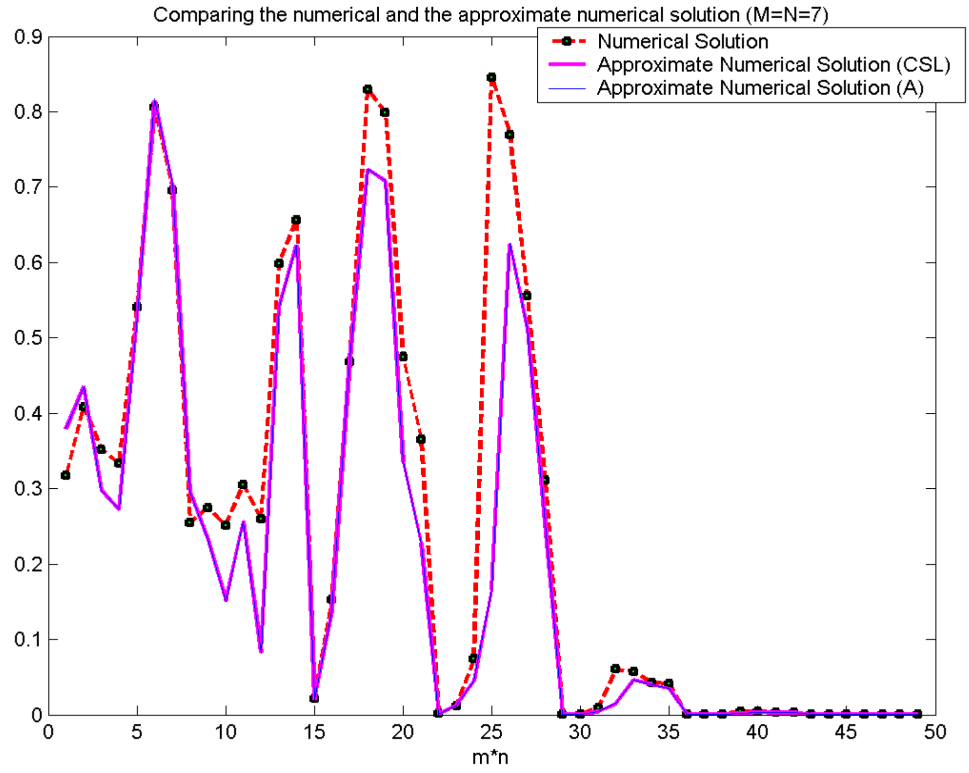


Figure 8.8: Approximated solution using the eigenvectors corresponding to the four smallest absolute eigenvalues of $M_{CSL}^{-1}A$ and A in the case of $M, N = 7$.

8.3 Eigenvalues and their Corresponding Coefficients

We have seen in Theorem 3.7 that

$$\mathbf{p} = c_1\mathbf{v}_1 + \ldots + c_n\mathbf{v}_n, \tag{8.4}$$

for a complex-symmetric matrix with distinct eigenvalues. The coefficients can be computed as follows:

$$c_i = \frac{\langle \mathbf{p}, \mathbf{v}_i \rangle}{\langle \mathbf{v}_i, \mathbf{v}_i \rangle} \in \mathbb{C}. \tag{8.5}$$

In fact, each $\mathbf{v}_i$ corresponds to a specific c_i. Since $\mathbf{v}_i$ is related to the eigenvalue λ_i, we can also say that λ_i corresponds with the coefficient $\mathbf{c}_i$. Now, we investigate whether there is a relation between λ_i and c_i for each i. Therefore, we reconsider and extend again Examples 1 and 2 and plot the following parameters against the coefficients c_i:

- real parts of the eigenvalues;
- imaginary parts of the eigenvalues;
- absolute values of the eigenvalues.

Before we give the results, one has to note that all $\mathbf{v}_i$ can be scaled by an $\alpha_i \in \mathbb{C}\backslash\{0\}$. We choose to normalize all $\mathbf{v}_i$, i.e., all $\mathbf{v}_i$ are forced to have length 1. This is important because of the following fact. If $\mathbf{v}_i$ is an eigenvector, then $\alpha_i\mathbf{v}_i, \alpha_i \in \mathbb{C}\backslash\{0\}$ is also an eigenvector. Then we obtain:

$$\frac{\langle \mathbf{p}, \alpha_i\mathbf{v}_i \rangle}{\langle \alpha_i\mathbf{v}_i, \alpha_i\mathbf{v}_i \rangle} = \frac{\alpha_i \langle \mathbf{p}, \mathbf{v}_i \rangle}{\alpha_i^2 \langle \mathbf{v}_i, \mathbf{v}_i \rangle} = \frac{1}{\alpha_i}\frac{\langle \mathbf{p}, \mathbf{v}_i \rangle}{\langle \mathbf{v}_i, \mathbf{v}_i \rangle} = \frac{1}{\alpha_i}c_i, \tag{8.6}$$

using Expression (8.5). In other words: the coefficients c_i are *dependent* on the choices of α_i. For a fair comparison, we choose the coefficients c_i such that each $\alpha_i\mathbf{v}_i$ has unit length, i.e.,

$$\alpha_i = \frac{1}{\sqrt{\langle \mathbf{v}_i, \mathbf{v}_i \rangle}}, \; \forall i = 1, 2, \ldots, n. \tag{8.7}$$

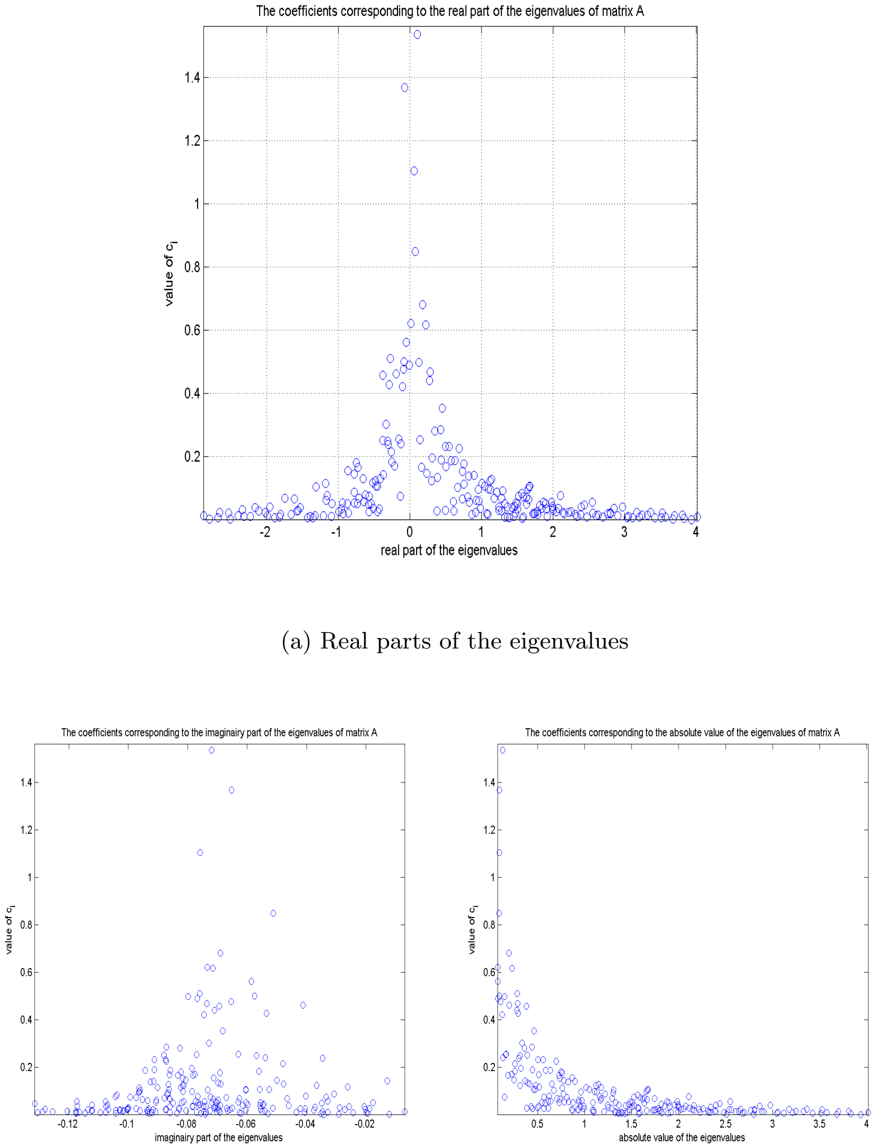
Now, the results of the two examples can be given, where Example 2 can be seen in **Appendix J**.

8.3.1 Second Extension to Example 1

In Figure 8.9, one can find the results where the eigenvalues are plotted against the coefficients c_i.

Considering the subplots of Figure 8.9, one concludes immediately that the absolute and the real parts of eigenvalues around zero correspond with relatively *large* coefficients. In other words: the eigenvectors corresponding to these 'bad' eigenvalues are the main components of the solution.

Note that the smallest real parts (in absolute sense) do not have large coefficients. In Figure 8.9(a), one can see that the large coefficients corresponds



(a) Real parts of the eigenvalues

(b) imaginary parts of the eigenvalues

(c) Absolute values of the eigenvalues

Figure 8.9: Real parts of the coefficients corresponding to the real, imaginary and absolute parts of the eigenvalues of A in the case of $M, N = 15$***.***

with real parts around zero, but there are also real parts around zero which have small coefficients. That means that the method described in the previous section can be improved by taking the (four) eigenvectors corresponding to the large coefficients instead of taking the eigenvectors corresponding to the 'bad' eigenvalues. However, in practice this is difficult to perform.

Observe further that the plot of the imaginary parts of the eigenvalues has a different structure. The imaginary parts near zero do not have the largest coefficients and hence, it is not used in further research.

Comparison of Real and Imaginary Parts and Absolute Values of the Coefficients

We take a look at the real and imaginary parts of the coefficients c_i, instead of at the absolute values. The results of the comparison for Example 1 can be found in Figure 8.10.

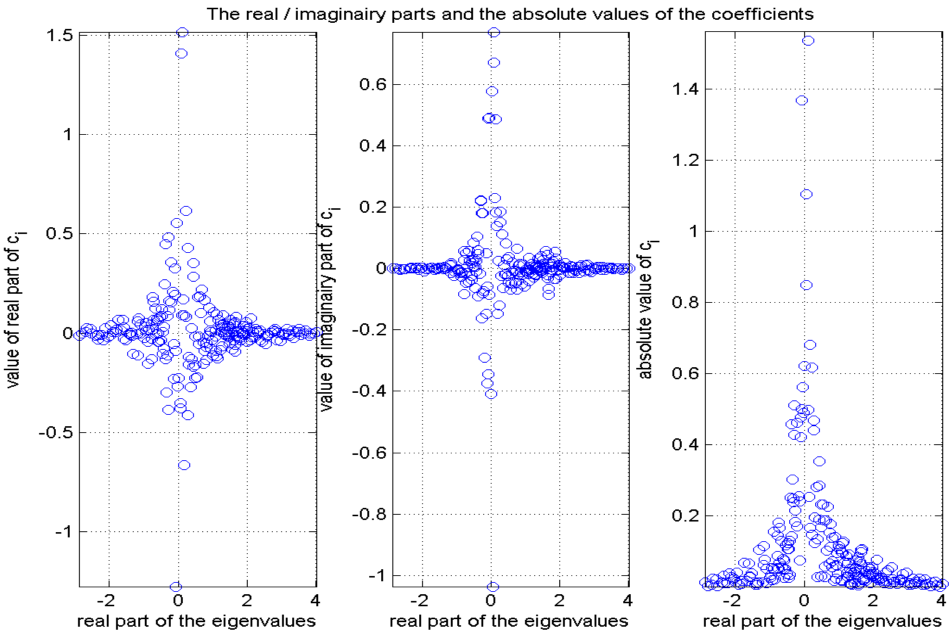


Figure 8.10: Real parts, imaginary parts and the absolute values of the coefficients corresponding to the real parts of the eigenvalues of A in the case of $M, N = 15$*.*

One can see in all subplots of Figure 8.10 that the eigenvalues around zero are the most important eigenvalues, since *large* coefficients in both real, imaginary and absolute sense are located in that region.

Number of Significant Coefficients

Next, we investigate the number k of significant coefficients with the help of the following definition:

$$P = \frac{\sum_{i=1}^{k} \hat{c}_i}{\sum_{i=1}^{n} \hat{c}_i}, \tag{8.8}$$

where $\langle \hat{c}_i \rangle_{i=1}^{n}$ is the sorted monotonically decreasing sequence of the absolute values of the coefficients of all c_i. Therefore: $\hat{c}_1 \geq \hat{c}_2 \geq \ldots \geq \hat{c}_n$.

Let m be a specific number such that $0 < m < 1$ (for instance: $m = 0.5, 0.75$ or 0.9). The question is: what is the minimum number k such that

$P > m$? In other words:

$$\text{find the minimum number } k \text{ such that } P = \frac{\sum_{i=1}^{k} \hat{c}_i}{\sum_{i=1}^{n} \hat{c}_i} > m. \tag{8.9}$$

Using MATLAB, the results are given in Table 8.1.

m	k	P
0.5	20	0.504
0.75	55	0.755
0.9	103	0.900

Table 8.1: Number of significant coefficients in absolute sense for matrix A in the case of $M, N = 15$.

Considering Table 8.1, one concludes that approximately 55 ‘bad’ eigenvectors are needed to represent 75% of the solution or only 103 ’bad’ eigenvectors are needed to represent 90% of the solution. Since there are in total $15 \times 15 = 225$ eigenvectors, these numbers are relatively low. However, in large practical problems, this number remains to be too large and hence, these ‘bad’ eigenvectors are too expensive to compute.

Conclusion

By reconsidering the examples, we conclude that the eigenvectors corresponding to the ‘bad’ eigenvalues determine mainly the approximation of the solution. This means that in future research, if we are looking for new preconditioners, we have to find one which can deal with these bad eigenvalues.

8.3.2 Coefficients after Adapting the Source Term

We vary the *location* and the *weight* of the point source in the source term f and look again at the coefficients corresponding to the (real part of the) eigenvalues, see Figure 8.11(a). This figure is analogous to Figure 8.10.

Different Location of the Source Term

In the next test runs, we move the point source vertically and horizontally with a quarter of the length of a specific direction of the domain and compute again the real/imaginary parts and the absolute values of the coefficients c_i corrsponding to the real part of the eigenvalues, see Figure 8.11.

One can see that there are obvious *differences* between the results of the subplots of Figure 8.11. However, in all figures one observes that the *large* values of c_i corresponds with the real parts around zero. Thus, considering this latter observation, the structure of the figures are all the *same*.

Scaling of the Source Term

If one takes $\alpha \cdot \mathbf{f}$, $\alpha > 0$ instead of the original $\mathbf{f}$, then the coefficients in Figure 8.11(a) are scaled with the same α. This can easily be seen: since $\mathbf{p} = \mathbf{A}^{-1}\mathbf{f}$ and therefore $\mathbf{A}^{-1}(\alpha\mathbf{f}) = \alpha\mathbf{A}^{-1}\mathbf{f} = \alpha\mathbf{p}$ holds, we obtain immediately:

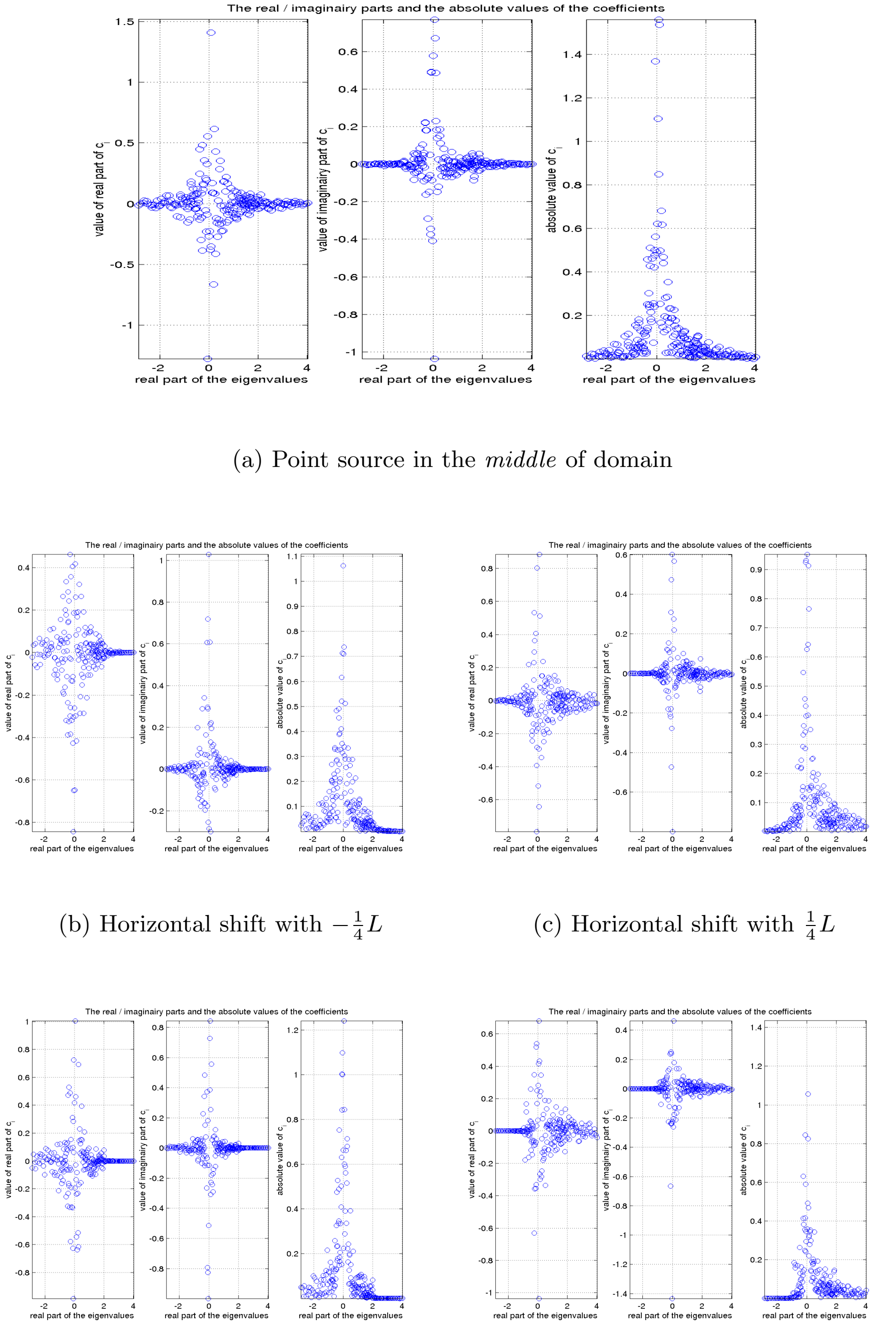
$$\alpha c_i = \frac{\langle \alpha\mathbf{p}, \mathbf{v}_i \rangle}{\langle \mathbf{v}_i, \mathbf{v}_i \rangle} = \frac{\left\langle \mathbf{A}^{-1}(\alpha\mathbf{f}), \mathbf{v}_i \right\rangle}{\langle \mathbf{v}_i, \mathbf{v}_i \rangle}. \tag{8.10}$$

An example of scaling the source term can be found in Figure 8.12, where $\alpha = 50$ is taken.

Conclusions

If one scales the source term, then the coefficients are also scaled by the same parameter. In this case, the plots have the same form.

Furthermore, if one moves the point source, the plots of the coefficients are somewhat different, but the main property holds,: the large coefficients correspond to the small real parts and also to the small absolute values of the eigenvalues.



(a) Point source in the *middle* of domain

(b) Horizontal shift with $-\frac{1}{4}L$

(c) Horizontal shift with $\frac{1}{4}L$

(d) Vertical shift with $\frac{1}{4}L$

(e) Vertical shift with $-\frac{1}{4}L$

Figure 8.11: Real parts, imaginary parts and the absolute values of the coefficients corresponding to the real parts of the eigenvalues of A in the case of $M, N = 15$ and point source in the middle, left, right, top and bottom of the domain, respectively.

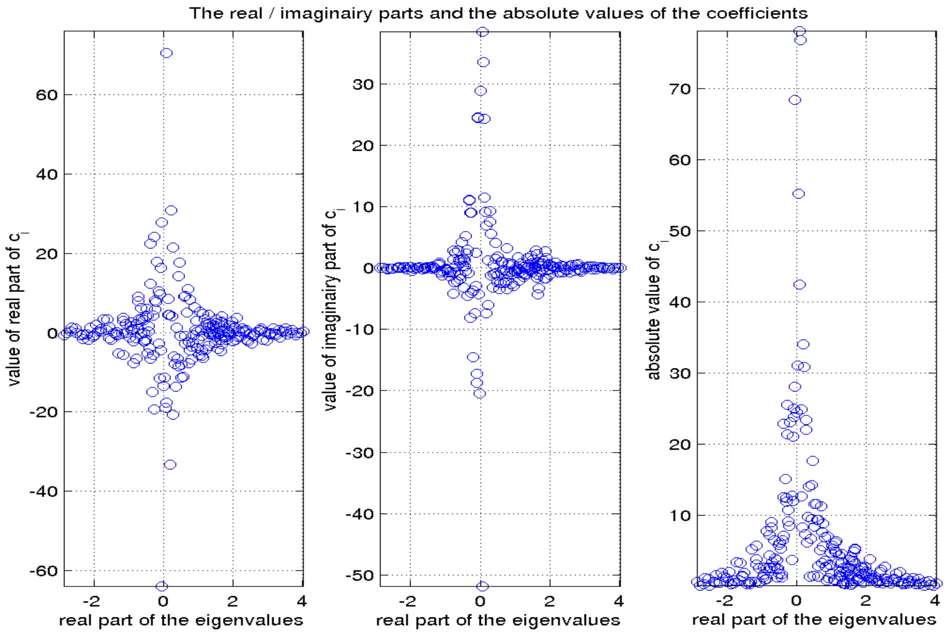


Figure 8.12: Real parts, imaginary parts and the absolute values of the coefficients corresponding to the real parts of the eigenvalues of $\boldsymbol{A}$ *in the case of* $M, N = 15$ *and source term* $50 \cdot \boldsymbol{f}$*.*

Chapter 9

Combined Preconditioners using SoV and CSL

In the previous chapter, we have noted that the 'bad' eigenvalues of the CSL preconditioned system corresponds to the 'bad' eigenvalues of $\mathbf{A}$, where 'bad' eigenvalues mean eigenvalues relatively close to zero. Now, when the few 'bad' eigenvalues of SoV correspond to *other* eigenvalues of $\mathbf{A}$, one could simply combine SoV and CSL to get rid of all bad eigenvalues of both preconditioners, which hopefully leads to fast convergence of the iterative method.

In this chapter, we try some combinations of SoV and CSL preconditioners, such that this leads to a new and 'powerful' preconditioner. Test problems are applied using the standard wedge model.

9.1 Modified SoV Preconditioner (SoV–σ)

In the test runs of Chapter 6, we have seen that an iterative method using the SoV preconditioner is faster in convergence than the CSL preconditioner. Furthermore, there are possibilities to combine CSL and SoV, because of the results of Chapter 8. Intuitively, it is easy to do a few iterations with CSL followed by the remaining iterations applying SoV. This is a first alternative for a combination of both preconditioners.

We denote this new preconditioner by SoV–σ where $\sigma \in \mathbb{N}$ is the number of starting iterations with the CSL preconditioner.

Results

The results of using SoV–σ with varying σ can be found in Table 9.1.

In this table, one can clearly see that SoV–σ leads to better results than SoV for some test problems and for appropriate choices of σ. But when SoV requires only a few (< 20) iterations, then SoV–σ does *not* improve the convergence speed.

It can also be observed that there is a kind of optimum of σ for the 'best' convergence. The optimal σ varies in each case of Table 9.1. Actually, the few starting iterations with CSL influences slightly the *starting vector* for SoV

Test Problem	M, N	SoV	SoV–1	SoV–2	SoV–3	SoV–4	SoV–5	SoV–10
(V)	15	78	79	75	71	72	72	**68**
	25	35	35	34	**33**	**33**	37	45
	35	32	**29**	32	32	33	35	41
(V+)	15	**18**	19	20	19	21	22	26
	25	**12**	**12**	13	14	15	16	21
	35	**10**	11	13	13	16	15	19
(V++)	15	69	**68**	69	70	74	69	81
	25	65	66	66	**60**	**60**	61	64
	35	**37**	38	39	**37**	39	41	45

Table 9.1: Number of iterations using Bi-CGSTAB in test problems in combination with the wedge model using the modified SoV-preconditioner (SoV–σ).

preconditioner. Therefore, the differences between the original SoV and the new SoV–σ are relatively small.

We end with the plots of the convergence behaviour of test problem (V) and $M, N = 15$, see Figure 9.1.

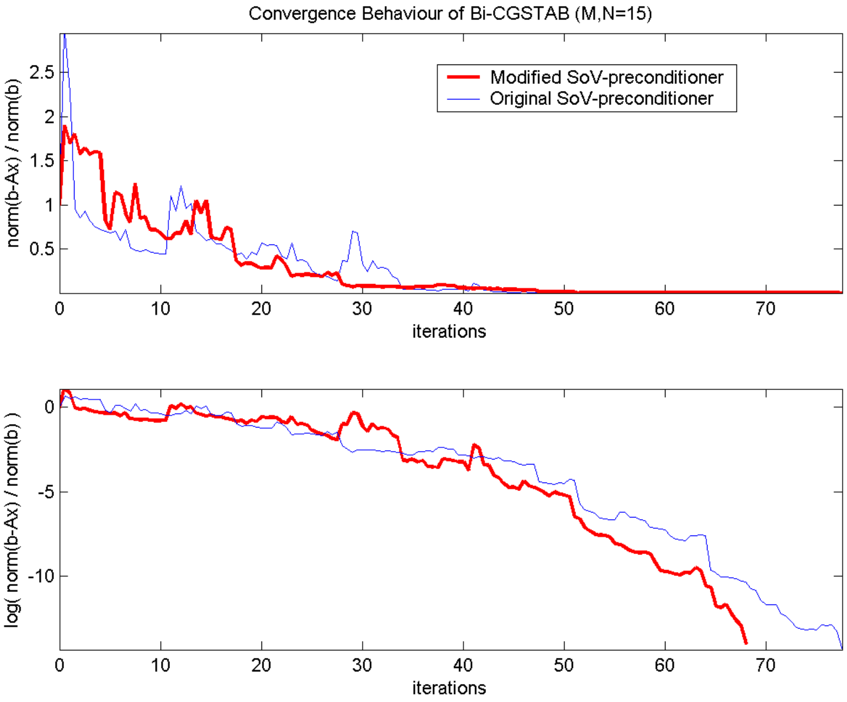


Figure 9.1: Convergence behaviour of Bi-CGSTAB with both the original SoV preconditioner (78 iterations) and the modified SoV–10 preconditioner (68 iterations) in test problem (V) with $M, N = 15$. Top subplot: the standard relative residuals during the iterations. Bottom subplot: the logarithms of these relative residuals.

Considering Figure 9.1, we conclude that the logarithmical residuals of the SoV–σ preconditioner have a stronger superlinear behavior compared to those of the standard SoV-preconditioner, but the differences are small.

Conclusion

We have seen that the SoV–σ preconditioner with particular σ leads to slightly better results than the SoV preconditioner.

However, the results of SoV–σ are not very promising, since the relative improvement with respect to SoV is less than 10% in most test runs. Problems with larger M and N may result in better performance. Furthermore, this combined preconditioner gives at least some perspective for other combinations of CSL and SoV, which may lead to new preconditioners with better convergence results, see the next sections where the *additive*, *multiplication* and *alternated* preconditioners are introduced.

9.2 Multiplication Preconditioner (MP)

An idea to combine the CSL and SoV preconditioners into one preconditioner is to define the following new preconditioners

$$\mathbf{M}_{MP1} = \mathbf{M}_{SoV}\mathbf{A}^{-1}\mathbf{M}_{CSL}, \tag{9.1}$$

and

$$\mathbf{M}_{MP2} = \mathbf{M}_{CSL}\mathbf{A}^{-1}\mathbf{M}_{SoV}, \tag{9.2}$$

where MP is an abbreviation of '*multiplication preconditioner*'. We apply these preconditioners in our iterative method Bi-CGSTAB. Note that approximately the same work is required for *each* iterate with $\mathbf{M}_{MP1}$ or $\mathbf{M}_{MP2}$ and *two* iterates with the original preconditioner SoV or CSL. This means that the MP is only efficient if it is at least twice as fast as CSL or SoV.

Results

The results of the test runs, using $\mathbf{M}_{MP1}$ and $\mathbf{M}_{MP2}$, can be found in Table 9.2.

Test Problem	M, N	CSL	MP1	MP2
(C++)	15	62	64	63
	25	59	62	63
	35	60	57	57
(V)	15	242	> 500	> 500
	25	160	> 500	> 500
	35	237	> 500	> 500

Table 9.2: Number of iterations of Bi-CGSTAB with both variants of the MP preconditioner.

Unfortunately, the results of both variants of MP are not satisfactory. MP1 and MP2 are slower than CSL in all runs, as can be seen in Table 9.2. This can also be illustrated by considering the eigenvalues of $\mathbf{M}_{MP1}^{-1}\mathbf{A}$ and $\mathbf{M}_{MP2}^{-1}\mathbf{A}$ and comparing these with $\mathbf{M}_{CSL}^{-1}\mathbf{A}$, see Figure 9.2.

The eigenvalues of $\mathbf{M}_{MP1}^{-1}\mathbf{A}$ or $\mathbf{M}_{MP2}^{-1}\mathbf{A}$ approach the eigenvalues of $\mathbf{M}_{CSL}^{-1}\mathbf{A}$, see Figure 9.2. They lie on the same kind of ellipse, except for a few eigenvalues which are scattered to the left and right of and above this ellipse. These scattered eigenvalues, especially the negative eigenvalues, cause the bad convergence of Bi-CGSTAB in combination with both MP1 and MP2.

Furthermore, considering the subplots of Figure 9.2 and the results of Table 9.2, we see that the eigenvalues of $\mathbf{M}_{MP1}^{-1}\mathbf{A}$ and $\mathbf{M}_{MP2}^{-1}\mathbf{A}$ and therefore, the number of iterations are more or less the same. This observation can even be proved, see below.

Theorem 9.1 *Let $\boldsymbol{M}_{MP1} = \boldsymbol{M}_{SoV}\boldsymbol{A}^{-1}\boldsymbol{M}_{CSL}$ and $\boldsymbol{M}_{MP2} = \boldsymbol{M}_{CSL}\boldsymbol{A}^{-1}\boldsymbol{M}_{SoV}$ where $\boldsymbol{A}, \boldsymbol{M}_{SoV}$ and $\boldsymbol{M}_{CSL}$ are arbitrary invertible matrices. Then the spectra of $\boldsymbol{M}_{MP1}^{-1}\boldsymbol{A}$ and $\boldsymbol{M}_{MP2}^{-1}\boldsymbol{A}$ are exactly the same.*

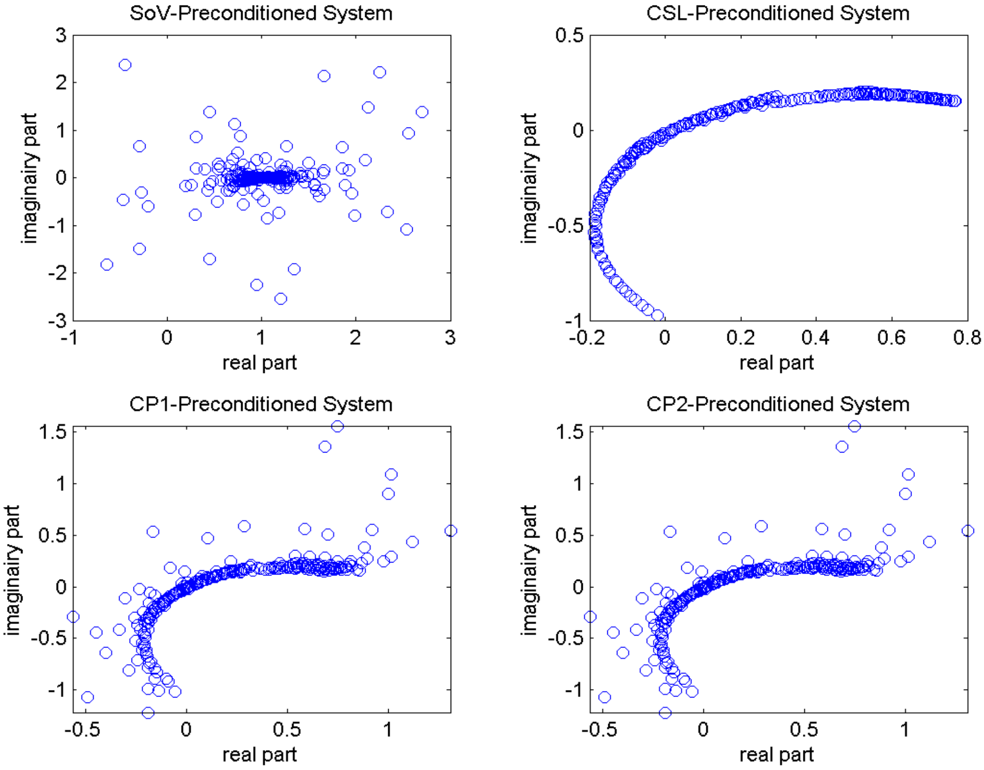


Figure 9.2: Eigenvalues of the systems $M_{SoV}^{-1}A$, $M_{CSL}^{-1}A$, $M_{MP1}^{-1}A$, $M_{MP2}^{-1}A$ in test problem (V) with the wedge model using $M, N = 15$.

Proof. Applying Theorem 3.8, we obtain that $\mathbf{M}_{SoV}^{-1}\mathbf{A}\mathbf{M}_{CSL}^{-1}\mathbf{A}$ has the same eigenvalue distribution as $\mathbf{M}_{CSL}^{-1}\mathbf{A}\mathbf{M}_{SoV}^{-1}\mathbf{A}$. Hence, the eigenvalues of $\mathbf{M}_{MP1}^{-1}\mathbf{A}$ and $\mathbf{M}_{MP2}^{-1}\mathbf{A}$ are the same.

□

9.3 Additive Preconditioner (SP)

In the previous section, we have seen the disappointing results of the MP. Another approach instead of using the MP is to take a new preconditioner of the form

$$\mathbf{M}_{SP} = \alpha \mathbf{M}_{SoV} + (1-\alpha)\mathbf{M}_{CSL}, \tag{9.3}$$

where $0 < \alpha < 1$. We call (9.3) the 'sum preconditioner' or the '*additive preconditioner*' (SP).

Results

Some results using $\mathbf{M}_{SP}$ can be found in Table 9.3.

				Additive Preconditioner			
Test Problem	M, N	SoV	CSL	$\alpha = 0.5$	$\alpha = 0.75$	$\alpha = 0.95$	$\alpha = 0.99$
	15	78	240	144	98	**65**	72
(V)	25	35	160	102	61	**34**	34
	35	**32**	237	128	78	36	34

Table 9.3: Number of iterations of Bi-CGSTAB with the SP preconditioner using various α.

The results of this new additive preconditioner SP are better than the multiplication preconditioner MP. However, also SP makes only sense if it is at least twice as fast as SoV or CSL, since SP requires computations with two preconditioners in each iterate. Therefore, the results found in Table 9.3 are still not satisfactory.

Note that the best results of the SP is obtained with a relatively large α, i.e., $0 \ll \alpha < 1$. This is the consequence of the fact that in all runs the original SoV performs a lot better than the original CSL, see columns 3 and 4 of Table 9.3.

The eigenvalues of SP with $\alpha = 0.5$ and $\alpha = 0.95$ can be found in Figure 9.3. It can be seen in this figure that the number of 'bad' eigenvalues in the SoV preconditioned system do not decrease in the SP preconditioned system.

There is another possibility to combine SoV and CSL preconditioner: Bi-CGSTAB using *alternately* the SoV and CSL preconditioner in each iterate. Unfortunately, due to the construction of Bi-CGSTAB, this does not give fast convergence as mentioned in **Appendix B**. However, methods like GCR are able to handle a so-called 'alternated' preconditioner, see the next sections. In fact, GCR applies (9.3) and chooses automatically the 'best' α in (9.3).

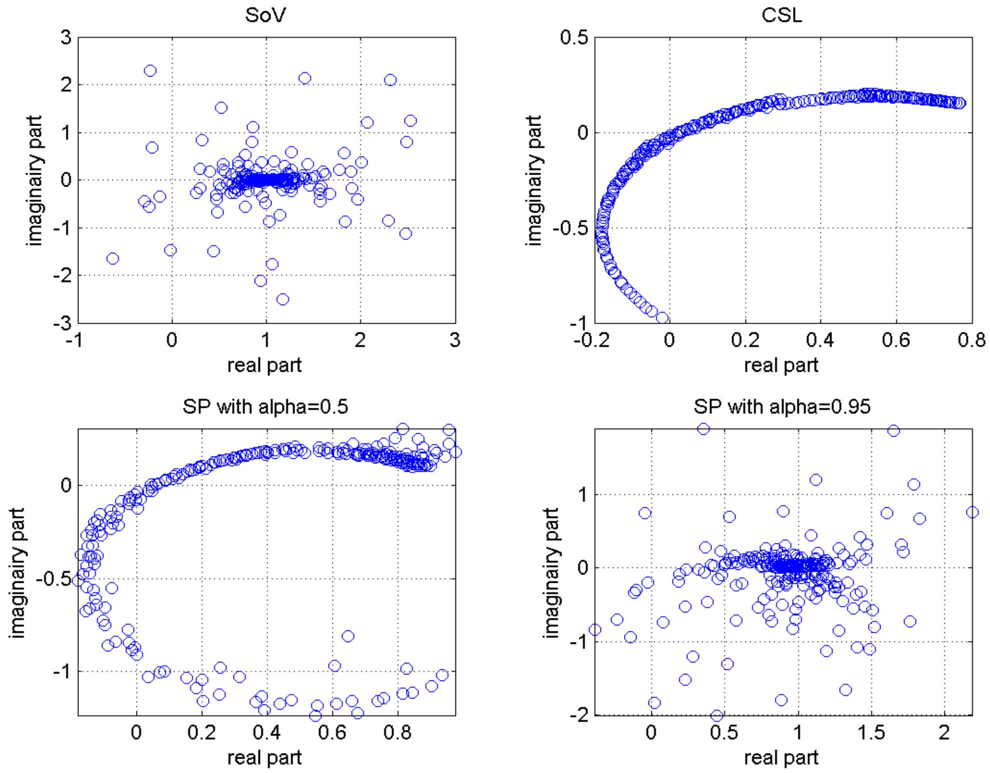


Figure 9.3: Eigenvalues of the systems $M_{SoV}^{-1}A$, $M_{CSL}^{-1}A$, $M_{SP}^{-1}A$, respectively, (with $\alpha = 0.5$ and $\alpha = 0.9$) in test problem (V) with the wedge model using $M, N = 15$.

9.4 Alternated Preconditioner (AP)

In this section, GCR is used as iterative method in combination with SoV, CSL and a new 'alternated preconditioner' (AP). In this AP preconditioner, we apply *alternately* SoV and CSL in each iterate. The results of some test runs can be found in Table 9.4.

Test Problem	M, N	SoV	CSL	AP
(V)	15	**62**	214	82
	25	**34**	166	58
	35	**32**	166	62
(E)	15	91	**84**	108
	25	**180**	491	266
	35	**164**	436	225

Table 9.4: Number of iterations with GCR using the SoV, CSL and the alternated preconditioner (AP), respectively, in test problem (V) in combination with the wedge model.

Considering Table 9.4, it can be noted that GCR in combination with the SoV preconditioner leads to the best results in most runs. More importantly, the AP shows better convergence than CSL in most cases and at the same time it is *worse* than SoV. A comparison between the number of iterations using the SP (Table 9.3) and the AP (Table 9.4) gives as a result that the SP seems to be a better preconditioner comparing to AP. However, each iterate of the SP requires computations with *two* preconditioners, whereas the AP needs one preconditioner in each iterate. From this point of view, both combined preconditioners are comparable to each other.

9.5 Alternated Preconditioner using full $\widetilde{\mathrm{K}}$ (AP-K)

In the AP, we have applied alternately SoV and CSL, but there are more alternatives available such as the AP with alternately SoV and a preconditioner based on $\widetilde{k}$. This can be motivated as follows. In Section 7.1, we have seen that the Helmholtz equation can be written as:

$$\Delta p - (k_x^2 + k_y^2)p - \widetilde{k}^2 p = f, \tag{9.4}$$

resulting in the linear system

$$\mathbf{A}_{SoV}\mathbf{p} - \widetilde{\mathbf{K}}\mathbf{p} = \mathbf{f}, \tag{9.5}$$

where $\widetilde{\mathbf{K}}$ and $\mathbf{A}_{SoV} = \hat{\mathbf{A}}$ as given in Expressions (7.13) and (7.60), respectively. One assumes $\widetilde{\mathbf{K}} = 0$ in the SoV preconditioner. Now, the idea of AP-K is to apply also $\widetilde{\mathbf{K}}$ in the preconditioner. Therefore, we take $\mathbf{M}_1 = \mathbf{A}_{SoV}$ and $\mathbf{M}_2 = \widetilde{\mathbf{K}}$ as preconditioners and apply them alternately in GCR. Note that $\widetilde{\mathbf{K}}$ is a diagonal matrix and thus, $\mathbf{M}_2$ is a preconditioner which is easy to use.

Results

In Table 9.5 one can find the results of some test runs using AP-K.

Test Problem	M, N	AP	AP-K
	15	82	103
(V)	25	58	69
	35	62	67
	15	108	171
(E)	25	266	357
	35	225	331

Table 9.5: Number of iterations with GCR using the alternated preconditioners AP and AP-K, respectively, in test problem (V) with the wedge model.

In Table 9.5, it can be seen that the AP-K does not work well. The AP-K is even less efficient than the AP preconditioner. However, considering the computational work, both combined preconditioners are comparable to each other, since the iterates with $\widetilde{\mathbf{K}}$ as preconditioner are relatively cheap.

9.6 Conclusions

We have tried some combined preconditioners in Bi-CGSTAB and GCR. These preconditioners fail in our test runs, comparing to the original SoV preconditioner, while the first combined preconditioner (SoV-σ) has given some perspective for finding better combined preconditioners.

The preconditioners SP, MP, AP and AP-K may work better, if the test problems are more complex by taking much more gridpoints. In these situations SoV and CSL show difficulties (see [15, 37, 52]) with as consequence that these combined preconditioners can be attractive. This is left for further research.

Chapter 10

Conclusions

As a result of the research, which has been described in the previous chapters, we can draw the following conclusions.

Theoretical Results using Linear Algebra

Assume $\mathbf{W}_R$ to be a right eigenvector matrix. Then, for an arbitrary complex diagonalizable matrix $\mathbf{A}$, we can choose a left eigenvector matrix $\mathbf{W}_L = \mathbf{W}_R^{-1}$ such that $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$ holds, where $\mathbf{I}$ is the unit matrix. If the matrix is even *complex-symmetric* and it has distinct eigenvalues, the choice $\mathbf{W}_L = \overline{\mathbf{W}}_R$ ensures the decomposition $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$.

Complex-symmetric matrices have *orthogonal* eigenvectors with respect to the *conjugate* inner product. However, the eigenvalues are *not* real, in contrast to Hermitian and real-symmetric matrices.

Let matrix $\mathbf{A}$ and the preconditioner $\mathbf{M}$ be symmetric and indefinite. Then, the eigenvalues of the system $\mathbf{M}^{-1}\mathbf{A}$ are, in general, *complex*. If $\mathbf{A}$ is *positive semi-definite* (PSD) instead of indefinite, then the eigenvalues of $\mathbf{M}^{-1}\mathbf{A}$ are *real*-valued.

The separation–of–variables (SoV) preconditioned matrix $\mathbf{M}_{SoV}^{-1}\mathbf{A}$, obtained with the discretized Helmholtz problem in combination with Dirichlet boundary conditions, has in general *complex* eigenvalues, while the *real* shifted Laplace preconditioned matrix $\mathbf{M}_{CSL}^{-1}\mathbf{A}$, obtained in the same way, has always *real* eigenvalues, which are easier to use in spectral analysis.

Properties of CSL and SoV

The complex shifted Laplace (CSL) preconditioner with $\alpha = 0$ and $\beta = 1$ is *not* efficient for all Helmholtz problems. For instance, in the case of the HP with *conjugate Sommerfeld absorbing conditions*, the conjugate CSL preconditioner with parameters $\alpha = 0$ and $\beta = -1$ is the most efficient choice.

Taking the maximum or minimum values, instead of averaging values, at the boundaries in the SoV preconditioner, results in approximately the *same* iterative behavior of Bi-CGSTAB.

Results of the Test Runs

In the test runs, we have that, by varying the gridsizes M and N, the number of iterations is approximately *constant* using the SoV preconditioner. Without preconditioner, the number of iterations *increases* by enlarging the gridsizes.

The SoV preconditioner works *better* than the CSL preconditioner in more or less all test runs.

Our small test problems are defined in such a way that the SoV preconditioner is very efficient. However, Plessix & Mulder [37] have shown that, for larger k and, therefore, for larger gridsizes, SoV is no longer efficient. It is unknown whether the results, obtained with our test problems, also hold for larger problems.

The failure of the SoV preconditioner is related to the contribution of $\widetilde{\mathbf{K}}$. It appears that if the number of iterations using SoV increases, then also the relative contribution of $\widetilde{\mathbf{K}}$ becomes larger.

Since we deal with complex vectors, the *order* of inner products in the algorithms of iterative methods (Bi-CGSTAB, GMRES and GCR) is important.

Improving the SoV preconditioner

Each wavenumber $k(x, y)$ can be decomposed into $k^2(x, y) = k_x^2(x) + k_y^2(y) + \tilde{k}^2(x, y)$. It appears that the $k_x(x)$ and the $k_y(y)$ in the SoV, proposed by Plessix & Mulder [37], are nearly optimal. Several attempts to choose other $k_x(x)$ and $k_y(y)$ in the SoV preconditioner do *not* lead to better performance of the iterative methods.

If we take $\widetilde{\mathbf{K}}_{diag}$ (resulting in a $\tilde{k}(x, y) \neq 0$) in the SoV preconditioner leads to somewhat better results, for relatively *small* gridsizes M and N.

Eigenvalue and Eigenvector Analysis

The eigenvectors belonging to the eigenvalues of $\mathbf{A}$ close to zero (i.e., the ‘bad’ eigenvalues) are the *main* components of the approximation of the solution $\mathbf{p}$ of $\mathbf{Ap} = \mathbf{f}$. Therefore, a ‘good’ preconditioner has to get rid of the bad eigenvalues of $\mathbf{A}$.

Bad eigenvalues of the CSL preconditioned system are *related* to the bad eigenvalues of the original matrix $\mathbf{A}$, while there is *no* implication that these correspond also to the bad eigenvalues of the SoV preconditioned system.

Construction of Combined Preconditioners

The combined preconditioner SoV–σ leads to somewhat *better* results than the CSL and SoV preconditioners

The other combined preconditioners (SP, MP, AP and AP-K) can be *faster* than CSL, but they are *slower* than SoV.

Chapter 11

Recommendations for Further Research

With regard to the conclusions in the previous chapter, we recommend the following actions to be taken or considered in future research.

We know that if we have an complex diagonalizable and complex-symmetric matrix with distinct eigenvalues, then the choice $\mathbf{W}_L = \overline{\mathbf{W}}_R$ ensures the decomposition $\mathbf{W}_L^H \mathbf{W}_R = \mathbf{I}$. Does the above statement also hold for the matrix *without* distinct eigenvalues.

In our test runs, we have restricted ourselves to small gridsizes ($M, N \leq 45$), while much larger gridsizes are required in practice. In Plessix & Mulder [37], it has been concluded that the SoV preconditioner is not efficient in complex models with these gridsizes and hence, the CSL preconditioner is more attractive in these models. One has to implement these larger problems and compare the behavior of the preconditioners once again.

More research is needed in choosing appropriate $k_x(x)$ and $k_y(y)$, which may improve the SoV preconditioner.

Several attempts failed to create a block-diagonal $\widetilde{\widetilde{\mathbf{K}}}$, which can be taken into account in the SoV preconditioner. We expect that it has to be *possible* to choose such an appropriate $\widetilde{\widetilde{\mathbf{K}}}$, which improves the SoV preconditioner a lot. In this kind of examinations, we need also test problems with larger gridsizes.

We know that eigenvectors, belonging to the small eigenvalues of the discretized HP, are the main components of the solution. One can try to prove and generalize this observation.

We have to find a preconditioner, which gets rid of the small eigenvalues of the original matrix $\mathbf{A}$, since we have shown that these are the most important eigenvalues of the system. [1]

[1] From this point of view, CSL is not a good preconditioner, since this preconditioner has the property that it does *not* get rid of the small eigenvalues of $\mathbf{A}$.

More research is needed to investigate the small eigenvalues of **A** and also of the CSL and SoV preconditioned systems. Considering the results, found with the combined preconditioners, it seems that the small eigenvalues of **A** are not all 'covered' by both CSL and SoV.

The defined combined preconditioners (SoV–σ, SP, MP, AP and AP-K) may show better performance in problems with larger gridsizes, since SoV is not always more efficient than CSL in these cases.

Other combined preconditioners applying existing preconditioners, like SoV, CSL, AILU, IC or multigrid preconditioners, may lead to better results.

In the results of the test problems we have seen that, by varying the gridsizes M and N, the number of iterations is approximately *constant*, using the SoV preconditioner. Therefore, the number of small eigenvalues does not increase by enlarging the gridsizes, which can also be seen in the spectral plots of Chapter 6.

From this point of view, '*successive refinement*' techniques can make sense. The idea behind these techniques is to solve the system $\mathbf{Ap} = \mathbf{f}$ exact on a *coarse* grid and prolongate this solution on the 'original' grid, such that this can be used as *starting vector* for the iterative method. The expectation is that this starting vector approaches the real solution very well, since it does not consist the influences of the small eigenvalues anymore. Therefore, we expect a fast convergence of the iterative method.

A start with these investigations is made. First results and more details about successive refinement can be found in **Appendix K**.

Bibliography

[1] Abrahamsson, L., Kreiss, H.O., *Numerical solution of the coupled mode equations in duct acoustics*, J. Comput. Phys., 111, pp. 1-14, 1994.

[2] Abramowitz, M., Stegun, I.A., (*Eds.*), *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, 9th Printing, New York: Dover, 1972.

[3] Achenbach, J.D., *Wave propagation in Elastic Solids*, North-Holland Publishing Company, 1973.

[4] Arnoldi, W. *The Principle of Minimized Iterations in the Solution of the Matrix Eigenvalue Problem*, Quart. Appl. Math., 9, pp. 17-29, 1951.

[5] Axelsson, O., *Iterative Solution Methods*, Cambridge University Press: Cambridge, 1994.

[6] Bayliss, A., Goldstein, C.I., Turkel, E., *An iterative method for Helmholtz equation*, J. Comput. Phys., 49, pp. 443-457, 1983.

[7] Bayliss, A., Goldstein, C.I., Turkel, E., *On accuracy conditions for the numerical computation of waves*, J. Comput. Phys., 59, pp. 396-404, 1985.

[8] Bayliss, A., Goldstein, C.I., Turkel, E., *The numerical solution of the Helmholtz equation for wave propagation problems in underwater acoustics*, Comput. Math. Appl., 11, pp. 655-665, 1985.

[9] Bayliss, A., Turkel, E., *Radiation boundary conditions for wave-like equations*, Comm. Pure Appl. Math, 33, pp. 707-725, 1980.

[10] Bayliss, A., Gunzburger, M., Turkel, E., *Boundary conditions for the numerical solution of elliptic equations in exterior regions*, SIAM, J. Appl. Math., 42, pp. 430-451, 1982.

[11] Boyce, W.E., DiPrima, R.C., *Elementary differential equations and boundary value problems*, 6th Edition, Wiley & Sons, 1997.

[12] Colton, J., Kress, R., *Inverse acoustic and electromagnetic scattering theory*, Springer-Verlag, Berlin-Heidelberg, 1998.

[13] Eisenstat, S.C., Elman, H.C., Schultz, M.H., *Variable iterative methods for nonsymmetric systems of linear equations*, SIAM, J. Num. Anal., 20, pp. 345-357, 1983.

[14] Erlangga, Y.A., *Some numerical aspects for solving sparse large linear systems derived from the Helmholtz equation*, report 02-12, TU Delft, 2002.

[15] Erlangga, Y.A., Vuik, C., Oosterlee, C.W., *On a class of preconditioners for solving the Helmholtz equation*, report 03-01, see `http://ta.twi.tudelft.nl/nw/users/vuik/papers/Erl03VO.pdf`, TU Delft, 2003.

[16] Ernst, O., Golub, G.H., *A domain decomposition approach to solving the Helmholtz equation with a radiation boundary condition*, in *Domain Decomposition Methods in Science and Engineering*, Quateroni et al. (editors), American Mathematical Society: Providence, RI, pp. 177-192, 1994.

[17] Fletcher, R., *Factorizing symmetric indefinite matrices*, Lin. Alg. and its Appl., 14, pp. 257-277, 1976.

[18] Gander, M.J., Nataf, F., *AILU for Helmholtz problems: a new preconditioner based on the analytic parabolic factorization*, J. Comp. Acoustics, 9, pp. 1499-1509, 2001.

[19] George, A., Liu, J.W., *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[20] Givoli, D., *Non-reflecting boundary conditions*, J. Comput. Phys., 94, pp. 1-29, 1991.

[21] Goldstein, C.I., *A finite element method for solving Helmholtz type equations in waveguides and other unbounded domains*, Math. Comp., 39 , pp. 309-324, 1982.

[22] Goldstein, C.I., *Multigrid preconditioners applied to the iterative solution of singularly perturbed elliptic boundary value problems and scattering problems*, Innovative Numerical Methods in Engineering, Proc. 4th Int. Symp., Atlanta/Ga., Springer-Verlag, 1986.

[23] Golub, G.H., Van Loan, C.F., *Matrix Computations*, Third Edition, The John Hopkins University Press, Baltimore, Maryland 21218, 1996.

[24] Graham, A., *Kronecker Products and Matrix Calculus With Applications*, Halsted Press, John Wiley and Sons, New York, 1981.

[25] Heikkola, E., Kuznetsoc, Y.A., Lipnikov, K.N., *Fictitious domain methods for the numerical solution of three-dimensional acoustic scattering problems*, J. Comp. Acoust., 7(3), pp. 161-183, 1999.

[26] Kan, J. van, Segal, A., *Numerieke methoden voor partiële differentiaalvergelijkingen*, Delftse uitgevers Maatschappij BV, 1993.

[27] Kim, J., Kim, D., *Absorbing boundary conditions*, report (see `http://kim.snu.ac.kr`), Seoul National University, 1997.

[28] Kim, J., *Absorbing Boundary Conditions for Wave Propagation in Viscoelastic Media*, J. Comput. Appl. Math., 76 , pp. 301-314, 1996.

[29] Laird, A.L., *Preconditioned iterative solution of the 2nd Helmholtz equation*, 1st year's report, Oxford University, St. Hugh's College, Oxford, 2001.

[30] Lay, D.C., *Linear algebra and its applications*, 3rd Edition, Addison Wesley Longman Inc., 2002.

[31] Made, M.M.M., *Incomplete factorization-based preconditionings for solving the Helmholtz equation*, Int. J. Numer. Meth. Engng., 50, pp. 1077-1101, 2001.

[32] Mitchell, A.R., Griffiths, D.F., *The finite difference method in partial differential equations*, Chichester: Wiley, 1994.

[33] Moore, T.G., Blaschak, J.G., Taflove, A., Kreigsmann, G.A., *Theory and application of radiation boundary operators*, IEEE Trans. Antennas and Propagation, 36, pp. 1797-1811, 1988.

[34] Nakos, G., Joyner, D., *Linear Algebra with Applications*, First Edition, Brooks/Cole Publishing Company, 1998.

[35] O'Nan, M., Enderton, H.B., *Linear Algebra*, Third Edition, San Diego, CA: Harcourt Brace Jovanovich, 1990.

[36] Ochmann, M., Mechel, F.P., *Analytical and Numerical Methods in Acoustics*, Springer-Verlag, 2002

[37] Plessix, R.E., Mulder, W.A., *Separation-of-variables as a preconditioner for an iterative Helmholtz solver*, Appl. Num. Math., 44(3), pp. 385-400, 2003.

[38] Rossi, T., Toivanen, J., *A parallel fast direct solver for block tridiagonal systems with separable matrix of arbitrary dimension*, SIAM, J. Sci. Comput., pp. 1778-1796, 1999.

[39] Saad, Y., *Iterative methods for sparse linear systems*, PWS Publishing Company, Boston, MA, 1996.

[40] Saad, Y., Schultz, M.H., *GMRES: A generalized minimal residual method for solving non-symmetric linear system*, SIAM, J. Sci. Statist. Comput., 7, pp. 856-869, 1986.

[41] Shewchuk, J.R., *An Introduction to the Conjugate Gradient Method without the Agonizing Pain*, Edition 1.25, Report, Carnegie Mellon University Pittsburgh, 1994.

[42] Sladek, S., Tanaka, M., Sladek, J., *Revised Helmholtz Integral Equation for Bodies Sitting on an InfinitePlane*, Transactions of the Japan Society for Computational Engineering and Science (JSCES), Paper No.20000039, 2002.

[43] Sleijpen, G.L.G., Fokkema, D.R., *BiCGstab(l) for linear equations involving unsymmetric matrices with complex spectrum.* Electron. Trans. Numer. Anal., 1(Sept.), pp. 11-32, 1993.

[44] Van der Sluis, A., *Conditioning, equilibration and pivoting in linear algebraic systems*, Numer. Math., 15, pp. 74-86, 1970.

[45] Smith, J.M., *Modern Numerical Integration Methods*, In *Mathematical Modeling and Digital Simulation*, 2nd Ed., New York: John Wiley, 1988.

[46] Sommerfeld, A., *Partial differential equations in physics*, Academic Press, New York, 1949.

[47] Sonneveld, P., *CGS: a fast Lanczos type solver for nonsymmetric linear systems*, SIAM, J. Sci. Stat. Comp., 10, pp. 36-53, 1989.

[48] Tang, J.M., *Numerical Aspects of Solving Linear Systems derived from Helmholtz's Problem*, Literature Report, see `http://ta.twi.tudelft.nl/nw/users/vuik/numanal/tang_eng.html`, 2004.

[49] Van der Vorst, H.A., *Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems*, SIAM, J. Sci. Statist. Comput., 13, pp. 631-644, 1992.

[50] Van der Vorst, H.A., Vuik, C., *The superlinear convergence behaviour of GMRES*, J. Comp. Appl. Math., 48, pp. 327-341, 1993.

[51] Vuik, C., *Numerieke methoden voor differentiaalvergelijkingen*, lecture-notes, wi2091/wi2092, TU Delft, 2000.

[52] Vuik, C., Erlangga, Y.A., Oosterlee, C.W., *Shifted Laplace preconditioners for the Helmholtz equations*, report 03-18, see `http://ta.twi.tudelft.nl/nw/users/vuik/papers/Vui03EO.pdf`, TU Delft, 2003.

[53] Vuik, C., *Numerical methods for large algebraic systems*, lecture-notes, see `http://ta.twi.tudelft.nl/nw/users/vuik/a228/wi4010_notes.pdf`, WI4010, TU Delft, 2004.

[54] Wesseling, P., *An Introduction to Multigrid Methods*, John Wiley & Sons, Chilchester, 1992.

Appendix A

Arnoldi's method

Arnoldi's procedure is an algorithm for building an orthonormal basis of the Krylov subspace $\mathcal{K}_m$. One variant of the algorithm is given below.

Arnoldi's algorithm

```
Choose a vector $\mathbf{v}_1$ of norm 1
**For** $j := 1, 2, \ldots, m$ **Do** :
    $h_{i,j} := (\mathbf{A}\mathbf{v}_j, \mathbf{v}_i)$ for $i = 1, 2, \ldots, j$
    $\mathbf{w}_j := \mathbf{A}\mathbf{v}_j - \sum_{i=1}^{j} h_{ij}\mathbf{v}_i$
    $h_{j+1,j} := ||\mathbf{w}_j||_2$
    $\mathbf{v}_{j+1} := \dfrac{\mathbf{w}_j}{h_{j+1,j}}$
**EndDo**
```

At each step, the algorithm multiplies the previous Arnoldi vector $\mathbf{v}_j$ by $\mathbf{A}$ and then orthonormalizes the resulting vector $\mathbf{w}_j$ against all previous vectors $\mathbf{v}_i$, by a standard Gram-Schmidt procedure.

One can prove the following propositions (see pp. 146–148 of Saad [39]).

Proposition A.1 *Assume that Arnoldi's algorithm does not stop before the m-th step. Then, the vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m$ form an orthonormal basis of the Krylov subspace*

$$\mathcal{K}_m = \text{span}\left\{\mathbf{v}_1, \mathbf{A}\mathbf{v}_1, \ldots, \mathbf{A}^{m-1}\mathbf{v}_1\right\}. \tag{A.1}$$

Proposition A.2 *Denote by $\boldsymbol{V}_m$ the $n \times m$ matrix with column vectors $\boldsymbol{v}_1, \ldots,$ $\boldsymbol{v}_m$. Denote by $\bar{\boldsymbol{H}}_m$ the $(m+1) \times m$ Hessenberg matrix whose nonzero entries $h_{i,j}$ are defined by Arnoldi's algorithm. Furthermore, denote by $e_m = \{0, 0, \ldots, 1\}^T$ and by $\boldsymbol{H}_m$ the matrix obtained from $\bar{\boldsymbol{H}}_m$ by deleting its last row. Then, the following relations hold:*

$$\begin{aligned} \mathbf{A}\mathbf{V}_m &= \mathbf{V}_m\mathbf{H}_m + \mathbf{w}_m e_m^T && \text{(A.2)} \\ &= \mathbf{V}_{m+1}\overline{\mathbf{H}}_m, && \text{(A.3)} \\ \mathbf{V}_m^T\mathbf{A}\mathbf{V}_m &= \mathbf{H}_m. && \text{(A.4)} \end{aligned}$$

Appendix B

Preconditioned Krylov Iterative Methods

The preconditioned Krylov iterative methods Bi-CGSTAB, GMRES and GCR are given below. The preconditioner is denoted by $\mathbf{M}$.

Algorithm B.1: Preconditioned Bi-CGSTAB

```
Compute $\mathbf{r}_0 := \mathbf{f} - \mathbf{A}\mathbf{p}_0$
Set $\mathbf{z}_0 := \mathbf{r}_0$, $\alpha_0 := \frac{\tilde{\mathbf{z}}_0}{(\mathbf{A}\mathbf{z}_0, \tilde{\mathbf{r}}_0)}$ and $\mathbf{s}_0 := \mathbf{r}_0 - \alpha_0 \mathbf{A}\mathbf{z}_0$
Choose $\tilde{\mathbf{r}}_0$ arbitrary

For $j := 0, 1, \ldots,$ until convergence Do :
    $\hat{\mathbf{z}}_j = \mathbf{M}^{-1}\mathbf{z}_j$
    $\hat{\mathbf{s}}_j = \mathbf{M}^{-1}\mathbf{s}_j$
    $\mathbf{w}_j := \mathbf{A}\hat{\mathbf{z}}_j$
    $\mathbf{v}_j := \mathbf{A}\hat{\mathbf{s}}_j$
    $\alpha_j := \frac{\tilde{\mathbf{z}}_j}{(\mathbf{w}_j, \tilde{\mathbf{r}}_0)}$
    $\mathbf{s}_j := \mathbf{r}_j - \alpha_j \mathbf{w}_j$
    $\omega_j := \frac{(\mathbf{v}_j, \mathbf{s}_j)}{(\mathbf{v}_j, \mathbf{v}_j)}$
    $\mathbf{p}_{j+1} := \mathbf{p}_j + \alpha_j \hat{\mathbf{z}}_j + \omega_j \hat{\mathbf{s}}_j$
    $\mathbf{r}_{j+1} := \mathbf{s}_j - \omega_j \mathbf{v}_j$
    $\beta_j := \frac{\alpha_j}{\omega_j} \frac{\rho_{j+1}}{\rho_j}$
    $\mathbf{z}_{j+1} := \mathbf{r}_j + \beta_j (\mathbf{z}_j - \omega_j \mathbf{w}_j)$
EndFor
```

Algorithm B.2: Preconditioned GMRES

```
Choose $\mathbf{x}_0$ and compute $\mathbf{r}_0 := \mathbf{f} - \mathbf{A}\mathbf{p}_0$, $\beta := ||\mathbf{r}_0||_2$ and $\mathbf{v}_1 := \mathbf{r}_0/\beta$
Define the $(m+1) \times m$ matrix $\mathbf{H}_m := \{h_{i,j}\}_{1 \le i \le m+1, 1 \le j \le m}$. Set $\mathbf{H}_m := 0$

For $j := 1, 2, \ldots,$ until convergence Do :
    $\mathbf{w}_j := \mathbf{M}^{-1}\mathbf{A}\mathbf{v}_j$

    For $i := 1, 2, \ldots, j$ Do :
        $h_{i,j} := (\mathbf{w}_j, \mathbf{v}_i)$
        $\mathbf{w}_j := \mathbf{w}_j - h_{ij}\mathbf{v}_i$
    EndFor

    $h_{j+1,j} := ||\mathbf{w}_j||_2$
    $\mathbf{v}_{j+1} := \frac{\mathbf{w}_j}{h_{j+1,j}}$

EndFor
Compute $\mathbf{y}_m := \arg\min_{\mathbf{y}} ||\beta e_1 - \bar{\mathbf{H}}_m \mathbf{y}||_2$
Compute $\mathbf{p}_m := \mathbf{p}_0 + \mathbf{V}_m \mathbf{y}_m$
```

Algorithm B.3: Preconditioned GCR

```
Choose $\mathbf{p}_0$ and compute $\mathbf{r}_0 := \mathbf{f} - \mathbf{A}\mathbf{p}_0$

For $j := 1, 2, \ldots,$ until convergence Do :
    $\mathbf{s}_j := \mathbf{M}^{-1}\mathbf{r}_{j-1}$
    $\mathbf{v}_j := \mathbf{A}\mathbf{s}_j$

    For $i := 1, 2, \ldots, j-1$ Do :
        $\alpha := (\mathbf{v}_j, \mathbf{v}_i)$
        $\mathbf{v}_j := \mathbf{v}_j - \alpha\mathbf{v}_i$,  $\mathbf{s}_j := \mathbf{s}_j - \alpha\mathbf{s}_i$
    EndFor

    $\mathbf{v}_j := \frac{\mathbf{v}_j}{||\mathbf{v}_j||_2}$,  $\mathbf{s}_j := \frac{\mathbf{s}_j}{||\mathbf{v}_j||_2}$

    $\mathbf{p}_j := \mathbf{p}_{j-1} + (\mathbf{r}_{j-1}, \mathbf{v}_j)\,\mathbf{s}_j$
    $\mathbf{r}_j := \mathbf{r}_{j-1} + (\mathbf{r}_{j-1}, \mathbf{v}_j)\,\mathbf{v}_j$

EndFor
```

Recall that $\mathbf{M}^{-1}$ is never computed in the implementation of the above algorithms. For instance, if we have to compute $\mathbf{s}_j := \mathbf{M}^{-1}\mathbf{r}_{j-1}$, then this is done by solving $\mathbf{M}\mathbf{s}_j = \mathbf{r}_{j-1}$ efficiently.

Moreover, note that $\mathbf{M}$ is required in determining the coefficients α_j in Algorithm B.1 and also in determining the coefficients $h_{i,j}$ in Algorithm B.2, whereas the coefficients α_j in Algorithm B.3 do not require $\mathbf{M}$. Therefore, in

GCR it is possible to apply alternately preconditioners $\mathbf{M}_1$ and $\mathbf{M}_2$ in the form of

$$\mathbf{M} = \begin{cases} \mathbf{M}_1, & \text{if } j = \text{even}; \\ \mathbf{M}_2, & \text{if } j = \text{odd}. \end{cases} \tag{B.1}$$

Appendix C

Diagonal and Incomplete Cholesky Preconditioners

In this appendix, we describe the diagonal (D) and the incomplete Cholesky (IC) preconditioners.

C.1 Diagonal Preconditioner

We transforms the original system in the following preconditioned system

$$\tilde{\mathbf{A}}\tilde{\mathbf{p}} = \tilde{\mathbf{f}}, \tag{C.1}$$

where $\tilde{\mathbf{A}} = \mathbf{P}^{-1}\mathbf{A}\mathbf{P}^{-T}, \mathbf{p} = \mathbf{P}^{-T}\tilde{\mathbf{p}}, \tilde{\mathbf{f}} = \mathbf{P}^{-1}\mathbf{f}$ and moreover, $\mathbf{P}$ is a non-singular matrix and $\mathbf{A}$ is *symmetric positive definite* matrix with dimensions $N \times N$.

A simple choice for $\mathbf{P}$ is a diagonal matrix with diagonal elements

$$\mathbf{P}_{ii} = \sqrt{\mathbf{A}_{ii}}, \quad \text{for } i = 1, 2, \ldots, N. \tag{C.2}$$

Then we can easily derive

$$\tilde{\mathbf{A}}_{ii} = \mathbf{P}_{ii}^{-1}\mathbf{A}_{ii}\mathbf{P}_{ii}^{-T} = 1, \quad \text{for } i = 1, 2, \ldots, N. \tag{C.3}$$

In Van der Sluis [44] it has been shown that this choice for $\mathbf{P}$ minimizes the condition number of $\tilde{\mathbf{A}}$, if $\mathbf{P}$ is restricted to be a *diagonal* matrix. For this preconditioner it is advantageous to apply CG to $\tilde{\mathbf{A}}\tilde{\mathbf{p}} = \tilde{\mathbf{f}}$, since $\tilde{\mathbf{A}}$ is easy to calculate.

However, in the HP we have a Hermitian and symmetric-complex matrix $\mathbf{A}$, instead of a symmetric positive definite matrix. More research is required to decide of the diagonal preconditioner can be used in this case.

C.2 Incomplete Choleski Factorization

Made [31] has introduced a new incomplete factorization based on a preconditioning technique, which consists in adding small perturbations to the *diagonal* entries of the real part of the matrix. In doing so, the real part is made positive definite, or less indefinite.

C.2.1 Idea of Incomplete Cholesky factorization

As a model problem, we consider again the large-scale linear system $\mathbf{Ap} = \mathbf{f}$, where $\mathbf{A}$ is *complex-symmetric* and the linear system is deduced from the Helmholtz problem. To solve this system with a *direct* method, one may factorize $\mathbf{A}$ as $\mathbf{A} = \tilde{\mathbf{L}}\tilde{\mathbf{L}}^T$ with $\tilde{\mathbf{L}}^T$ being lower triangular, and solves successively two triangular systems. This is known as the *Cholesky factorization.* Even if $\mathbf{A}$ is sparse, $\tilde{\mathbf{L}}^T$ is generally *less sparse* due to fill-in. This makes direct methods both memory and time consuming for N fairly large, as in real-life scientific or industrial problems.

A common remedy consists in ignoring certain fill-in entries, which yields an *incomplete* factorization preconditioning matrix $\mathbf{f} = \mathbf{LL}^T$. There are two basic strategies for accepting or discarding fill-in:

- by *level of fill-in* (or by position). The level '$\text{lev}(l_{i,j})$' of the coefficient $l_{k,i}$ of matrix $\mathbf{L}$ is defined by Saad, see section 10.3.3 of [39],

  1. *initialization*:

  $$\text{lev}(l_{i,j}) := \begin{cases} 0 & \text{if } l_{i,j} \neq 0 \text{ or } k = i, \\ \infty & \text{otherwise,} \end{cases} \tag{C.4}$$

  2. *factorization*:

  $$\text{lev}(l_{i,j}) = \min\{\text{lev}(l_{i,j}), \text{lev}(l_{i,k}) + \text{lev}(l_{k,j}) + 1\}, \tag{C.5}$$

  which is updated each time in line 5 of the algorithm of Gauss elimination (IKJ-variant), see [39].

  The set $\mathcal{D}$ of fill-in entries to be discarded is taken as

  $$\mathcal{D} = \{(i,j) \mid \text{lev}(l_{i,j} > \xi)\}, \tag{C.6}$$

  where the integer ξ denotes a user specified maximal fill-in level;

- by (numerical) *value.* Fill-in is ignored if it is 'too small' with respect to some prescribed tolerance.

Dual approaches that combine ingredients from both structural and numerical strategies are also used. The choice of both ξ and the drop tolerance depends, among other things, on the problem at hand and the available workspace. Several variants of the basic incomplete factorization have been designed, ranging from *modified* methods in which the discarded fill-in entries are added to the diagonal, to more sophisticated multilevel versions that use multigrid like (re)numbering strategies, see for instance Axelsson [5].

C.2.2 IC Preconditioners

Made has used six variants of preconditioners based on the incomplete Cholesky factorization, i.e.,

1. **IC**: the standard incomplete Cholesky applied to $\mathbf{A}$;
2. **MIC**: the standard modified incomplete Cholesky applied to $\mathbf{A}$ (see for instance Section 5.1 of Vuik [53] for more details about this method);
3. $\mathbf{IC}_0$: IC applied to $\mathbf{A}_0 \equiv \mathrm{Re}(\mathbf{A}) + \mathbf{Q} \quad (\gamma = 1)$;
4. $\widetilde{\mathbf{IC}}_0$: IC applied to $\widetilde{\mathbf{A}} \equiv \mathbf{A}_0 + i\ \mathrm{Im}(\mathbf{A}) = \mathbf{A} + \mathbf{Q}$;
5. $\mathbf{MIC}_0$: MIC applied to $\mathbf{A}_0 \quad (\gamma = 1)$;
6. $\widetilde{\mathbf{MIC}}$: MIC applied to $\widetilde{\mathbf{A}}$,

where $\mathbf{Q}$ stands for the diagonal matrix, whose diagonal entries q_{ii} are defined by

$$q_{ii} = -\gamma \min\{0, \mathrm{Re}((\mathbf{Ae})_i)\}, \tag{C.7}$$

with $\mathbf{e}$ the all-one vector and γ a given real parameter. In fact, $\mathbf{Ae}$ is the vector with the sum of the rows of $\mathbf{A}$. It can be proved, using spectral analysis (see section 4.2 of [31]), that $\mathbf{A}_0$ is a diagonal perturbation of the *Hermitian* part of the system matrix, while with $\tilde{\mathbf{A}}$, the same diagonal perturbation is added to the *whole* matrix.

In numerical experiments, using restarted GMRES (section 5.2 of [31]), we can see that varying values for, respectively, wavenumber k, stepsize h, number of fill-levels, parameter γ etcetera lead to different results. It appears that there is no best preconditioner in most situations, although often $\widetilde{\mathbf{MIC}}$ is the better one.

In this thesis we apply the preconditioner based on the standard incomplete Cholesky (IC) where only the real part of $\mathbf{A}$, as in $\mathbf{Ap} = \mathbf{f}$, is taken.

Appendix D

Comparing Eigenvalues of 1-D HP using DBC and CS-ABC

In Chapter 3 of Tang [48], we have considered the 1-D HP with both *absorbing* and *Dirichlet* boundary conditions. We have seen in both situations that the *real* parts of the eigenvalues of matrix $\mathbf{A}$ were approximately identical, which is favorable in iterative methods. If it can be shown that the HP with arbitrary boundary conditions leads to the same eigenvalue distributions in the real parts, then it will be sufficient to look at the Dirichlet HP in further research of convergence of iterative methods. The analysis of a Dirichlet problem is easier than a problem with absorbing conditions, since there are for instance no imaginary components in the problem.

The observation that the real part of the spectrum of $\mathbf{A}$ is as good as independent of the boundary conditions is suprisingly. In the example below, we see that the eigenvalues of a small matrix *do* depend on the choice of boundary conditions.

Example

Consider the following small matrix:

$$\mathbf{A} = \begin{bmatrix} 2+\alpha & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2+\alpha \end{bmatrix}, \quad \alpha \in \mathbb{C}, \tag{D.1}$$

which resembles the 1-D HP by taking $k = 0$ and $\Delta x = 1$. The parameter $\alpha \in \mathbb{C}$ in (D.1) can be determined by the corresponding boundary conditions of the HP. If $\alpha = 0$, then we deal with *Dirichlet* conditions. Moreover, if $\alpha = 1$, then we have absorbing conditions, since

$$\alpha = \frac{1}{\Delta x^2(1 + ik\Delta x)} = 1, \tag{D.2}$$

see also Section 2.5.

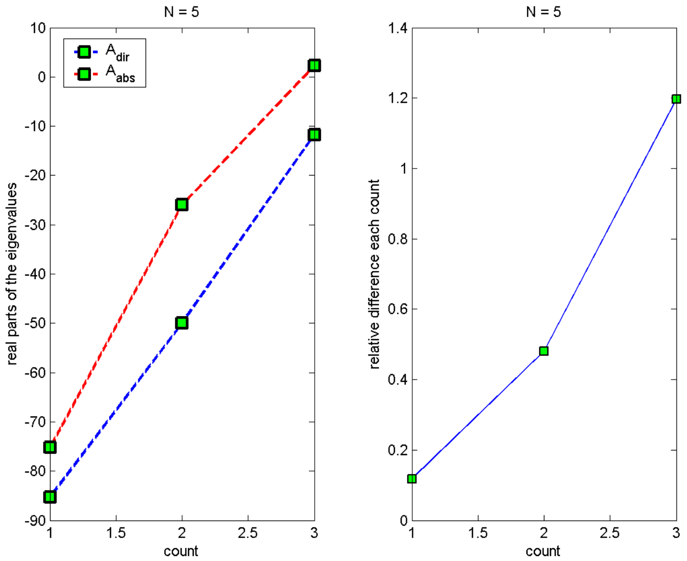


Figure D.1: ***Real part of the eigenvalues of the matrix A of HP with*** $k = 1$ ***and*** $\Delta x = 1/5$***. The line for the absorbing problem lies above the line for the Dirichlet problem.***

Next, the eigenvalues of system (D.1) turns out to be

$$\lambda_1 = 2 + \alpha, \quad \lambda_{2,3} = 2 + \frac{1}{2}\alpha \pm \sqrt{\frac{1}{4}\alpha^2 + 2}. \tag{D.3}$$

Therefore, it can be immediately seen that the eigenvalues depends on α, thus on the boundary conditions of HP.

□

D.1 MATLAB Tests

In MATLAB, we implement matrix $\mathbf{A}$ of the HP with both Dirichlet and absorbing conditions, denoted by $\mathbf{A}_{dir}$ and $\mathbf{A}_{abs}$, respectively. We have used a constant wavenumber: $k = 1$. The plots of the corresponding eigenvalues for $N = 5$ can be found in the left subplot of Figure D.1. [1]

We define the relative diffence δ_i for each count of the two sorted sets in the following way:

$$\delta_i = \frac{|(\lambda_{dir})_i - (\lambda_{abs})_i|}{|(\lambda_{abs})_i|}, \tag{D.4}$$

where $(\lambda_{dir})_i$ and $(\lambda_{abs})_i$ denote the i-th element of the sorted set of eigenvalues of the Dirichlet and absorbing problem, respectively. This difference between the real parts of the eigenvalues can be seen in the right plot of Figure D.1.

[1] In fact, we have $N - 2$ points since the boundary points are not considered in the matrix.

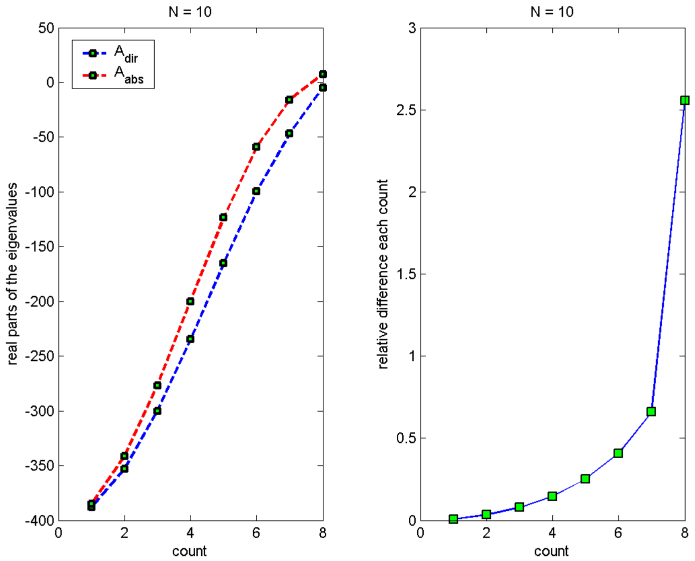


Figure D.2: Eigenvalues of the matrix A of HP with $k = 1$ ***and*** $\Delta x = 1/10$.

We have repeated the tests for various values of N, see Figures D.2 - D.4, where in each figure we have shown on the left, the plot of the real part of the eigenvalues, and, on the right, the relative difference δ_i.

In these figures, we observe the following:

- if N is relatively small, the eigenvalues of the Dirichlet and absorbing problem are obviously different. The differences δ_i are relatively large. If we increase N, then the relative differences become smaller, which can also be seen in the left plots;

- the right plots of δ_i behave as an exponential function. This can be expected, since the eigenvalues tend to zero in each left plot, if we increase N. Thus, small differences between the eigenvalues around zero can give large values of δ_i for i approaching N;

- the disadvantage in iterative methods, that some real parts of the eigenvalues are close to zero, can be cancelled if the imaginary parts are sufficiently far from zero. In our problem, this is exactly the case, see Figure D.5 and also Figure 3.6 of Tang [48]. With the help of these figures, we conclude that $|\lambda_{\min}| < \min|\lambda|$, where $\lambda_{\min}$ denotes the minimum value of the *real* part of the eigenvalues (i.e., $\lambda_{\min} \equiv \min \mathcal{R}(\lambda)$) and $\min|\lambda|$ denotes the minimum of the *modulus* of the (complex) eigenvalues of $\mathbf{A}_{abs}$. However, later on we shall see that there is only a slight difference using $|\lambda_{\min}|$ or $\min|\lambda|$ in our spectral analysis.

Consider now

$$K_1(\mathbf{A}) = |\lambda_{\max} - \lambda_{\min}|, \qquad \text{(D.5)}$$

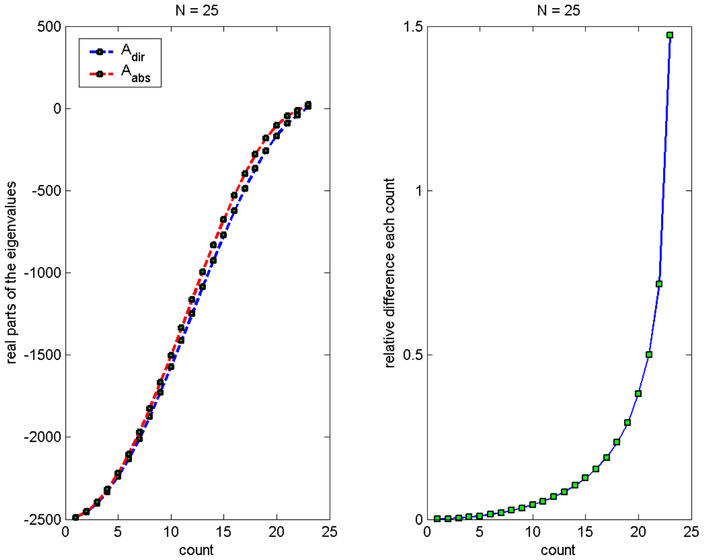


Figure D.3: Eigenvalues of the matrix A of HP with $k = 1$ *and* $\Delta x = 1/25$*.*

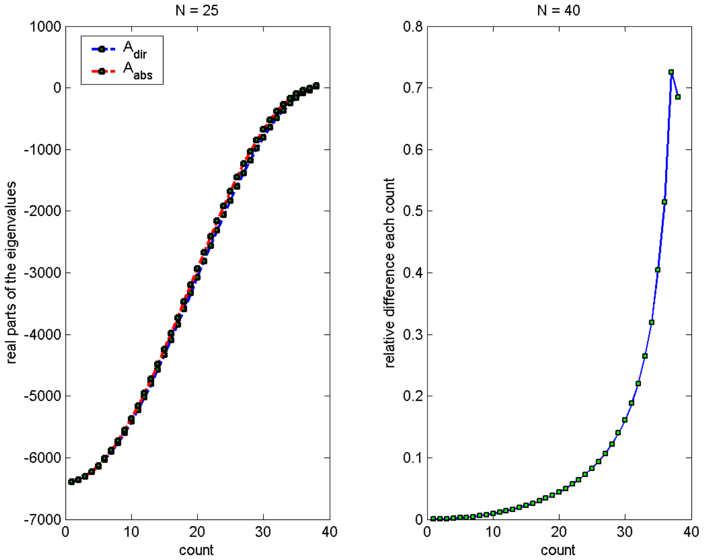


Figure D.4: Eigenvalues of the matrix A of HP with $k = 1$ *and* $\Delta x = 1/40$*.*

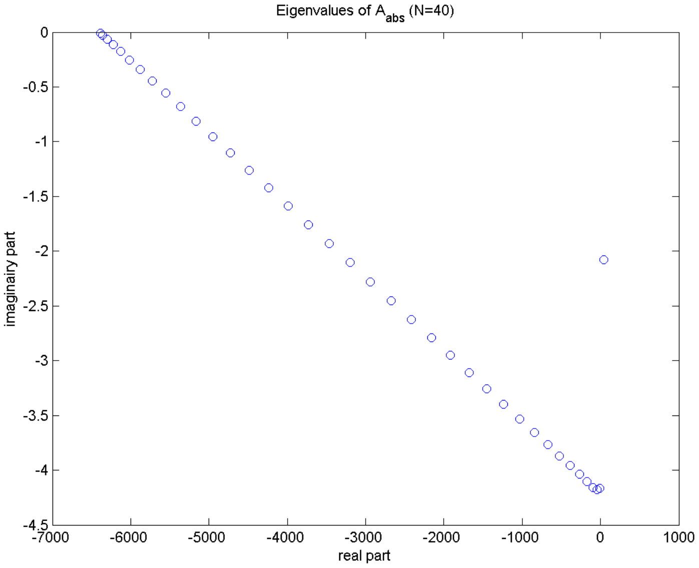


Figure D.5: The real and imaginary parts of the eigenvalues of the matrix A of HP with $k = 1$ and $\Delta x = 1/40$.

and

$$K_2(\mathbf{A}) = \frac{|\lambda_{\max}|}{|\lambda_{\min}|}, \tag{D.6}$$

and finally

$$K_3(\mathbf{A}) = \frac{|\lambda_{\max}|}{\min|\lambda|}, \tag{D.7}$$

where $\mathbf{A}$ is $\mathbf{A}_{dir}$ or $\mathbf{A}_{abs}$. Furthermore, $\lambda_{\max}$ and $\lambda_{\min}$ are the *maximum* and *minimum* real part of the eigenvalues of $\mathbf{A}$, respectively. Then, in fact $K_1(\mathbf{A})$ denotes the range of the eigenvalues, $K_2(\mathbf{A})$ is the condition number for symmetric and real $\mathbf{A}$ and $K_3(\mathbf{A})$ is a kind of the condition number for complex matrices. In Table D.1, we compute K_1 and K_2 for Figures D.1 to D.4.

N	$K_1(\mathbf{A}_{Dir})$	$K_1(\mathbf{A}_{abs})$	$K_2(\mathbf{A}_{Dir})$	$K_2(\mathbf{A}_{abs})$	$K_3(\mathbf{A}_{Dir})$	$K_3(\mathbf{A}_{abs})$
5	73.5	77.6	7.2	32.8	7.2	19.2
10	383.1	392.4	79.9	56.9	79.9	48
25	2499	2511	271	205	271	193
40	6412	6427	285	561	285	527

Table D.1: Outcomes of K_1, K_2 and K_3 in the Dirichlet and absorbing problem with $N = 5, 10, 25, 40$.

In Table D.2, the quantities δ_1,δ_2 and δ_3 denote the relative differences between $K_1(A_{Dir})$ and $K_1(A_{abs})$, $K_2(A_{Dir})$ and $K_2(A_{abs})$, $K_3(A_{Dir})$ and $K_3(A_{abs})$, respectively.

N	δ_1	δ_2	δ_3
5	0.05	3.6	1.7
10	0.05	0.29	0.39
25	0.00	0.24	0.29
40	0.00	0.97	0.85

Table D.2: The relative differences of K_1, K_2 and K_3 between the two problems.

One concludes that the relative differences δ_1 are in all situations very small ($\delta_1 \leq 0.05$), i.e., the range of the real parts of the eigenvalues are approximately the same in both the Dirichlet and the absorbing problem. However, the relative differences δ_2 and δ_3 are comparatively large between both problems, which can lead to different convergence results in iterative methods. This is the result of the bad location of $|\lambda_{\min}|$ in all situations, which is very close to zero. We end with the remark that δ_2 and δ_3 resemble each other, so in these problems there is no preference for using K_2 or K_3.

In the next section we give some analytical evidence of the fact that δ_1 is generally very small.

D.2 1-D Continuous HP

We consider the *continuous* 1-D HP with *Dirichlet*, *Neumann* and *absorbing* boundary conditions, respectively, and we determine the analytical eigenvalues. In general, the eigenvalues of the discrete HP converges to these analytical eigenvalues (when $M, N \to \infty$).

D.2.1 Dirichlet Conditions

In subsection 3.3.1 of [48], we have defined the 1-D HP with Dirichlet conditions, i.e.,

$$\begin{cases} \left(-\frac{\mathrm{d}^2}{\mathrm{d}x^2} - k^2\right) p(x) = f, \qquad x \in (0,1), \\ \\ p(0) = p(1) = 0. \end{cases} \tag{D.8}$$

Then, the eigenvalue equation is

$$\begin{cases} \left(-\frac{\mathrm{d}^2}{\mathrm{d}x^2} - k^2\right) \phi(x) = \lambda\phi(x), \qquad x \in (0,1), \\ \\ \phi(0) = \phi(1) = 0, \end{cases} \tag{D.9}$$

where λ and $\phi(x)$ are the eigenvalue and eigenfunction of system (D.8), respectively. The solution of (D.9) is

$$\phi(x) = c_1 \cos(\sqrt{\lambda + k^2}x) + c_2 \sin(\sqrt{\lambda + k^2}x). \tag{D.10}$$

Since we are not interested in the trivial solution, we have obtained the equation

$$\sin(\sqrt{\lambda + k^2}) = 0. \tag{D.11}$$

This has given us immediately the eigenvalues

$$\lambda_n = (n\pi)^2 - k^2, \tag{D.12}$$

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

D.2.2 Neumann Conditions

Consider now the Neumann problem:

$$\begin{cases} \left(-\dfrac{\mathrm{d}^2}{\mathrm{d}x^2} - k^2\right) p(x) = f, & x \in (0,1), \\ \\ \dfrac{\mathrm{d}}{\mathrm{d}x} p(0) = \dfrac{\mathrm{d}}{\mathrm{d}x} p(1) = 0. \end{cases} \tag{D.13}$$

The eigenvalues obey

$$\begin{cases} \left(-\dfrac{\mathrm{d}^2}{\mathrm{d}x^2} - k^2\right) \phi(x) = \lambda\phi(x), & x \in (0,1), \\ \\ \phi'(0) = \phi'(1) = 0. \end{cases} \tag{D.14}$$

The solution of (D.14) is

$$\phi(x) = c_1 \cos(\alpha x) + c_2 \sin(\alpha x), \tag{D.15}$$

where $\alpha = \sqrt{\lambda + k^2}$ Therefore,

$$\phi'(x) = -c_1\alpha \sin(\alpha x) + c_2\alpha \cos(\alpha x). \tag{D.16}$$

Using the boundary conditions, it can be derived that $c_2 = 0$ and

$$-c_1\alpha \sin(\alpha) = 0. \tag{D.17}$$

We are not interested in the trivial solution. Hence, we obtain the eigenvalues

$$\lambda_n = (n\pi)^2 - k^2, \tag{D.18}$$

for $n = 0, 1, 2, \ldots$. Note that these are almost the *same* eigenvalues as in the Dirichlet problem (see previous subsection). In the Neumann problem, we have one extra eigenvalue ($\lambda_0 = -k^2$) relative to the previous problem. Furthermore, using (D.9) and (D.14), it is easy to see that the eigenfunctions $\phi(x)$ are *different* for both problems.

D.2.3 Absorbing Conditions

The eigenvalue problem with absorbing conditions reads

$$\left(-\frac{\mathrm{d}^2}{\mathrm{d}x^2} - k^2\right)\phi(x) = \lambda\phi(x), \qquad x \in (0,1), \tag{D.19}$$

with boundary conditions

$$\begin{aligned} \left(-\frac{\mathrm{d}}{\mathrm{d}x} + ik\right)\phi(x) &= 0, \qquad x = 0, \\ \left(\frac{\mathrm{d}}{\mathrm{d}x} + ik\right)\phi(x) &= 0, \qquad x = 1. \end{aligned} \tag{D.20}$$

Now, we apply the same method as in the previous sections to this absorbing problem. We have the following corresponding eigenvalue problem:

$$\left\{ \begin{array}{rcl} -\phi''(x) - (k^2 - \lambda)\phi(x) & = & 0, \quad x \in (0,1), \\ -\phi'(0) + ik\phi(0) & = & 0, \\ \phi'(1) + ik\phi(1) & = & 0. \end{array} \right. \tag{D.21}$$

The general solution can be found:

$$\phi(x) = c_1 \cos(\alpha x) + c_2 \sin(\alpha x), \tag{D.22}$$

where $\alpha = \sqrt{\lambda + k^2}$. Therefore,

$$\phi'(x) = -c_1\alpha \sin(\alpha x) + c_2\alpha \cos(\alpha x). \tag{D.23}$$

Substituting (D.22) and (D.23) into the absorbing conditions of (D.21) leads to

$$\left\{ \begin{array}{rcl} c_2\alpha & = & ikc_1 \\ (c_1\alpha - ikc_2)\sin\alpha & = & (c_2\alpha + ikc_1)\cos\alpha \end{array} \right. \tag{D.24}$$

Combining the two expressions in (D.24) gives us immediately

$$\boxed{\tan\alpha = \frac{2ik\alpha}{\alpha^2 + k^2}} \tag{D.25}$$

with $i^2 = -1, k \in \mathbb{R}^+$ and $\alpha = \sqrt{\lambda + k^2}$, where λ still needs to be determined.

Solution

In this subsection we try the find a solution of (D.25).

Note that it is obvious that α can not be real-valued, since we then obtain only the trivial solution $\alpha = 0$ from (D.25). Moreover, α can not be pure complex, i.e., there exist no α of the form $\alpha = bi$ where $b \in \mathbb{R}$, because then we obtain

$$\tan(bi) = \tilde{b}i, \;\; \tilde{b} = \tanh b \in \mathbb{R}, \tag{D.26}$$

on the left hand side of (D.25). In expression (D.26), we have used the fact that

$$\tan x = i\frac{e^{-ix} - e^{ix}}{e^{-ix} + e^{ix}}, \tag{D.27}$$

and therefore

$$\tan bi = i\frac{e^{b} - e^{-b}}{e^{b} + e^{-b}} = \tanh bi = \tilde{b}i, \tag{D.28}$$

where $\tilde{b} = \tanh b = \frac{e^b - e^{-b}}{e^b + e^{-b}}$. Then, $\tan bi$ is indeed purely imaginary and $-2kb/(k^2 - b^2)$ on the right hand side of (D.25) is purely real-valued. This leads to the trivial solution $\alpha = 0$.

As a consequence, we have to look for solutions of (D.25) in the form

$$\alpha = a + bi, \tag{D.29}$$

with $a, b \in \mathbb{R}\backslash\{0\}$. First, we substitute (D.29) into the left-hand-side of expression (D.25) and we use the fact that

$$\tan(x + y) = \frac{\tan x + \tan y}{1 - \tan x \tan y}, \quad x, y \in \mathbb{C}. \tag{D.30}$$

Then we obtain the following:

$$\begin{aligned} \tan(a + bi) &= \frac{\tan a + \tan bi}{1 - \tan a \tan bi} = \frac{\tan a + i \tanh b}{1 - i \tan a \tanh b} \\ &= \frac{(\tan a + i \tanh b)(1 + i \tan a \tanh b)}{1 + \tan^2 a \tanh^2 b} \\ &= u + iv, \end{aligned} \tag{D.31}$$

where

$$u = \frac{\tan a(1 - \tanh^2 b)}{1 + \tan^2 a \tanh^2 b}, \quad v = \frac{\tanh b(\tan^2 a + 1)}{1 + \tan^2 a \tanh^2 b}. \tag{D.32}$$

Next, we treat the right-hand-side of expression (D.25):

$$\begin{aligned} \frac{2ik\alpha}{\alpha^2 + k^2} &= \frac{2ik(a + bi)}{(a + bi)^2 + k^2} = \frac{-2bk + 2aki}{(a^2 - b^2 + 2abi)} \\ &= \frac{(-2bk + 2aki)(a^2 - b^2 - 2abi)}{(a^2 - b^2 + 2abi)(a^2 - b^2 - 2abi)} \\ &= d + ei, \end{aligned} \tag{D.33}$$

where

$$d = \frac{4a^2bk - 2bk(a^2 - b^2 + k^2)}{(a^2 - b^2 + k^2)^2 + 4a^2b^2}, \quad e = \frac{2ak(a^2 - b^2 + k^2) + 4ab^2k}{(a^2 - b^2 + k^2)^2 + 4a^2b^2}. \tag{D.34}$$

Now, we rewrite equation (D.25) as a system of two equations

$$\left\{\begin{array}{lcl} \dfrac{\tan a(1-\tanh^2 b)}{1+\tan^2 a\tanh^2 b} & = & \dfrac{4a^2bk-2bk(a^2-b^2+k^2)}{(a^2-b^2+k^2)^2+4a^2b^2}; \\ \dfrac{\tanh b(\tan^2 a+1)}{1+\tan^2 a\tanh^2 b} & = & \dfrac{2ak(a^2-b^2+k^2)+4ab^2k}{(a^2-b^2+k^2)^2+4a^2b^2}, \end{array}\right. \tag{D.35}$$

combining (D.32) and (D.34). However, this new system (D.35) is not easy to solve.

Furthermore, if we have solved this system, then we need to solve the next problem:

$$\sqrt{c+di} = a+bi, \quad a,b,c,d \in \mathbb{R}, \tag{D.36}$$

where $\lambda + k^2 = c + di$. This leads to:

$$\left\{\begin{array}{lcl} a & = & c^2-d^2; \\ b & = & 2cd. \end{array}\right. \tag{D.37}$$

which gives the following solutions for c and d:

$$\left\{\begin{array}{lcl} c_{1,2,3,4} & = & \pm\dfrac{2b}{-2a\pm\sqrt{a^2+b^2}}; \\ d_{1,2,3,4} & = & \pm\dfrac{1}{2}\sqrt{-2a\pm\sqrt{a^2+b^2}}. \end{array}\right. \tag{D.38}$$

The problem is obviously too complex to solve analytically. We finish here our analysis.

D.2.4 Conclusion

The eigenvalues of both the continuous Dirichlet and Neumann problem are

$$\lambda_n = (n\pi)^2 - k^2, \tag{D.39}$$

where we note that also $\lambda_0 = -k^2$, is also an eigenvalue in the Neumann problem.

Moreover, the eigenvalues of the the continuous absorbing problem are too difficult to determine for us. If we assume the eigenvalues of this problem to be approximately the same as the Dirichlet problem, then we see that enlarging k leads to a shift of the eigenvalues in the direction of the negative axis. Moreover, enlarging the gridsizes M, N leads to a better approximation of the eigenvalues in (D.39), in general. Since λ_n in (D.39) are not bounded, the eigenvalues of the discrete problem increase if M, N also become larger. This is also the reason why the number of iterations in the HP increases in the iterative method (without preconditioner), when we enlarge the gridsizes M, N.

Appendix E

Examples illustrating the SoV Technique

Example E.1

Consider $k(x,y) = \sqrt{xy}$ at $(x,y) \in (0,1)^2$, then $k^2(x,y) = xy$. Moreover, using Theorem 5.1 we obtain:

$$\begin{cases} k_x^2(x) &= \int_0^1 xy \,\mathrm{d}y &= \frac{1}{2}x; \\ k_y^2(y) &= \int_0^1 \left(xy - \frac{1}{2}x\right) \,\mathrm{d}x &= \frac{1}{2}y - \frac{1}{4}; \\ \tilde{k}^2 &= k^2(x,y) - k_x^2(x) - k_y^2(y) &= xy - \frac{1}{2}x - \frac{1}{2}y + \frac{1}{4}. \end{cases}$$

This decomposition satisfy the two conditions of Theorem 5.1:

$$\begin{cases} \int_0^1 \tilde{k}^2(x,y) \,\mathrm{d}x &= \int_0^1 xy - \frac{1}{2}x - \frac{1}{2}y + \frac{1}{4} \,\mathrm{d}x = 0, \\ \int_0^1 \tilde{k}^2(x,y) \,\mathrm{d}y &= \int_0^1 xy - \frac{1}{2}x - \frac{1}{2}y + \frac{1}{4} \,\mathrm{d}y = 0. \end{cases}$$

□

Example E.2

Consider matrix $\mathbf{B}$, as in (7.46), with $M = 3$ and $N = 3$ and write

$$\mathbf{B} = \left[\begin{array}{ccc|ccc|ccc} b_1 & & & u_1 & & & & & \\ & b_2 & & & u_2 & & & & \\ & & b_3 & & & u_3 & & & \\ \hline l_1 & & & b_4 & & & u_4 & & \\ & l_2 & & & b_5 & & & u_5 & \\ & & l_3 & & & b_6 & & & u_6 \\ \hline & & & l_4 & & & b_7 & & \\ & & & & l_5 & & & b_8 & \\ & & & & & l_6 & & & b_9 \end{array}\right], \tag{E.1}$$

with b_i, l_i and u_i to be the non-zero elements of matrix $\mathbf{B}$. Moreover,

$$\mathbf{P} = \mathbf{P}^T = \left[\begin{array}{ccc|ccc|ccc}
1 & & & & & & & & \\
 & & & 1 & & & & & \\
 & & & & & & 1 & & \\
\hline
 & 1 & & & & & & & \\
 & & & & 1 & & & & \\
 & & & & & & & 1 & \\
\hline
 & & 1 & & & & & & \\
 & & & & & 1 & & & \\
 & & & & & & & & 1
\end{array}\right].$$

We first compute $\mathbf{P}^T\mathbf{B}$:

$$\mathbf{P}^T\mathbf{B} = \left[\begin{array}{ccc|ccc|ccc}
b_1 & & & u_1 & & & & & \\
l_1 & & & b_4 & & & u_4 & & \\
 & & & l_4 & & & b_7 & & \\
\hline
 & b_2 & & & u_2 & & & & \\
 & l_2 & & & b_5 & & & u_5 & \\
 & & & & l_5 & & & b_8 & \\
\hline
 & & b_3 & & & u_3 & & & \\
 & & l_3 & & & b_6 & & & u_6 \\
 & & & & & l_6 & & & b_9
\end{array}\right]. \tag{E.2}$$

Comparing the matrices (E.1) and (E.2), we see that the elements of $\mathbf{P}^T\mathbf{B}$ are *vertically* moved with respect to $\mathbf{B}$.

Now we can finally determine $\mathbf{D}$:

$$\mathbf{D} = (\mathbf{P}^T\mathbf{B})\mathbf{P} = \left[\begin{array}{ccc|ccc|ccc}
b_1 & u_1 & & & & & & & \\
l_1 & b_4 & u_4 & & & & & & \\
 & l_4 & b_7 & & & & & & \\
\hline
 & & & b_2 & u_2 & & & & \\
 & & & l_2 & b_5 & u_5 & & & \\
 & & & & l_5 & b_8 & & & \\
\hline
 & & & & & & b_3 & u_3 & \\
 & & & & & & l_3 & b_6 & u_6 \\
 & & & & & & & l_6 & b_9
\end{array}\right]. \tag{E.3}$$

Observe that now, comparing matrices (E.2) and (E.3), the elements of $\mathbf{D}$ are *horizontally* moved with respect to $\mathbf{P}^T\mathbf{B}$, .

Indeed, matrix $\mathbf{D}$ has a *block diagonal* structure with blocks with sizes $N \times N = 3 \times 3$. We can notice further that the diagonal elements of each diagonal block of $\mathbf{D}$ are the collection of elements of the same position of all diagonal blocks of $\mathbf{B}$. For instance, the diagonal elements b_1, b_4, b_7 are the diagonal elements of the first block of $\mathbf{D}$, which are positioned in the same position as the diagonal blocks of $\mathbf{B}$.

A last remark can be made about l_i and u_i. The index i has only been added to investigate the new position of the elements, while all l_i and u_i are the *same* elements. The latter statement is an immediate consequence of (7.27) and (E.3).

□

Example E.3

Consider again the problem with $M, N = 3$. Denote

$$\mathbf{I}_y \otimes \mathbf{W}_L^H = \left[\begin{array}{ccc|ccc|ccc} w_1 & w_2 & w_3 & & & & & & \\ w_4 & w_5 & w_6 & & & & & & \\ w_7 & w_8 & w_9 & & & & & & \\ \hline & & & w_{10} & w_{11} & w_{12} & & & \\ & & & w_{13} & w_{14} & w_{15} & & & \\ & & & w_{16} & w_{17} & w_{18} & & & \\ \hline & & & & & & w_{19} & w_{20} & w_{21} \\ & & & & & & w_{22} & w_{23} & w_{24} \\ & & & & & & w_{25} & w_{26} & w_{27} \end{array}\right].$$

Notice that $w_i = w_{i+9} = w_{i+18}$ for all $i = 1, \ldots, 9$. Now, we can compute matrix $\mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)$:

$$\mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H) = \left[\begin{array}{ccc|ccc|ccc} w_1 & w_2 & w_3 & & & & & & \\ & & & w_{10} & w_{11} & w_{12} & & & \\ & & & & & & w_{19} & w_{20} & w_{21} \\ \hline w_4 & w_5 & w_6 & & & & & & \\ & & & w_{13} & w_{14} & w_{15} & & & \\ & & & & & & w_{22} & w_{23} & w_{24} \\ \hline w_7 & w_8 & w_9 & & & & & & \\ & & w_{16} & w_{17} & w_{18} & & & & \\ & & & & & & w_{25} & w_{26} & w_{27} \end{array}\right].$$

Note that the elements of $(\mathbf{I}_y \otimes \mathbf{W}_L^H)$ have vertically been shifted. Finally, we can determine vector $\mathbf{v}$:

$$\begin{aligned} \mathbf{v} &= (\mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H))\,\mathbf{p} \\ &= \left(\begin{array}{c} w_1p_1 + w_2p_2 + w_3p_3 \\ w_{10}p_4 + w_{11}p_5 + w_{12}p_6 \\ w_{19}p_7 + w_{20}p_8 + w_{21}p_9 \\ \hline w_4p_1 + w_5p_2 + w_6p_3 \\ w_{13}p_4 + w_{15}p_5 + w_{16}p_6 \\ w_{22}p_7 + w_{23}p_8 + w_{24}p_9 \\ \hline w_7p_1 + w_8p_2 + w_9p_3 \\ w_{16}p_4 + w_{17}p_5 + w_{18}p_6 \\ w_{25}p_7 + w_{26}p_8 + w_{27}p_9 \end{array}\right) = \left(\begin{array}{c} w_1p_1 + w_2p_2 + w_3p_3 \\ w_1p_4 + w_2p_5 + w_3p_6 \\ w_1p_7 + w_2p_8 + w_3p_9 \\ \hline w_4p_1 + w_5p_2 + w_6p_3 \\ w_4p_4 + w_5p_5 + w_6p_6 \\ w_4p_7 + w_5p_8 + w_6p_9 \\ \hline w_7p_1 + w_8p_2 + w_9p_3 \\ w_7p_4 + w_8p_5 + w_9p_6 \\ w_7p_7 + w_8p_8 + w_9p_9 \end{array}\right) = \begin{pmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{pmatrix}, \end{aligned}$$

where $\mathbf{p} = (p_1, p_2, \ldots, p_9)^T$ and

$$\mathbf{v}_1 = \begin{pmatrix} w_1p_1 + w_2p_2 + w_3p_3 \\ w_1p_4 + w_2p_5 + w_3p_6 \\ w_1p_7 + w_2p_8 + w_3p_9 \end{pmatrix}, \quad \mathbf{v}_2 = \begin{pmatrix} w_4p_1 + w_5p_2 + w_6p_3 \\ w_4p_4 + w_5p_5 + w_6p_6 \\ w_4p_7 + w_5p_8 + w_6p_9 \end{pmatrix},$$

$$\mathbf{v}_3 = \begin{pmatrix} w_7p_1 + w_8p_2 + w_9p_3 \\ w_7p_4 + w_8p_5 + w_9p_6 \\ w_7p_7 + w_8p_8 + w_9p_9 \end{pmatrix}.$$

We can determine $\mathbf{g} = \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)\mathbf{f} = (\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3)^T$ in an analogous way.

□

Example E.4

Consider again the problem with $M = N = 3$, matrix $\mathbf{D}$ as defined in Example E.2, matrices $\mathbf{v}$ and $\mathbf{g}$ as in Example E.3. Then, we have to solve three problems of size 3:

$$\begin{cases} \mathbf{D}_1\mathbf{v}_1 & = & \mathbf{g}_1 \\ \mathbf{D}_2\mathbf{v}_2 & = & \mathbf{g}_2 \\ \mathbf{D}_3\mathbf{v}_3 & = & \mathbf{g}_3 \end{cases}$$

where

$$\mathbf{D}_1 = \begin{bmatrix} b_1 & u_1 & \\ l_1 & b_4 & u_4 \\ & l_4 & b_7 \end{bmatrix}, \ \mathbf{D}_2 = \begin{bmatrix} b_2 & u_2 & \\ l_2 & b_5 & u_5 \\ & l_5 & b_8 \end{bmatrix}, \ \mathbf{D}_3 = \begin{bmatrix} b_3 & u_3 & \\ l_3 & b_6 & u_6 \\ & l_6 & b_9 \end{bmatrix}.$$

□

Example E.5

Consider a problem with $M, N = 3$. Since $\mathbf{I}_y \otimes \mathbf{W}_L^H$ and $\mathbf{I}_y \otimes \mathbf{W}_R$ have a block structure and $\widetilde{\mathbf{K}}$ is diagonal, we can write

$$(\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \widetilde{\mathbf{K}}\ (\mathbf{I}_y \otimes \mathbf{W}_R) = \left[\begin{array}{ccc|ccc|ccc} x_1 & x_2 & x_3 & & & & & & \\ x_4 & x_5 & x_6 & & & & & & \\ x_7 & x_8 & x_9 & & & & & & \\ \hline & & & x_{10} & x_{11} & x_{12} & & & \\ & & & x_{13} & x_{14} & x_{15} & & & \\ & & & x_{16} & x_{17} & x_{18} & & & \\ \hline & & & & & & x_{19} & x_{20} & x_{21} \\ & & & & & & x_{22} & x_{23} & x_{24} \\ & & & & & & x_{25} & x_{26} & x_{27} \end{array}\right].$$

Then, we obtain

$$\begin{aligned}\widetilde{\widetilde{\mathbf{K}}} &= \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \widetilde{\mathbf{K}}\ (\mathbf{I}_y \otimes \mathbf{W}_R)\mathbf{P} \\ &= \left[\begin{array}{ccc|ccc|ccc} x_1 & & & x_2 & & & x_3 & & \\ & x_{10} & & & x_{11} & & & x_{12} & \\ & & x_{19} & & & x_{20} & & & x_{21} \\ \hline x_4 & & & x_5 & & & x_6 & & \\ & x_{13} & & & x_{14} & & & x_{15} & \\ & & x_{22} & & & x_{23} & & & x_{24} \\ \hline x_7 & & & x_8 & & & x_9 & & \\ & x_{16} & & & x_{17} & & & x_{18} & \\ & & x_{25} & & & x_{26} & & & x_{27} \end{array}\right].\end{aligned}$$

□

Example E.6

In Example E.2 with $M, N = 3$, we have seen that:

$$\widetilde{\mathbf{K}}' = \left[\begin{array}{ccc|ccc|ccc} x_1 & x_2 & x_3 & & & & & & \\ x_4 & x_5 & x_6 & & & & & & \\ x_7 & x_8 & x_9 & & & & & & \\ \hline & & & x_{10} & x_{11} & x_{12} & & & \\ & & & x_{13} & x_{14} & x_{15} & & & \\ & & & x_{16} & x_{17} & x_{18} & & & \\ \hline & & & & & & x_{19} & x_{20} & x_{21} \\ & & & & & & x_{22} & x_{23} & x_{24} \\ & & & & & & x_{25} & x_{26} & x_{27} \end{array}\right],$$

since $\widetilde{\mathbf{K}}' = (\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \widetilde{\mathbf{K}}\ (\mathbf{I}_y \otimes \mathbf{W}_R)$. Therefore, we obtain:

$$\begin{aligned}\widetilde{\widetilde{\mathbf{K}}} &= \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \widetilde{\mathbf{K}}\ (\mathbf{I}_y \otimes \mathbf{W}_R)\mathbf{P} \\ &= \left[\begin{array}{ccc|ccc|ccc} x_1 & & & x_2 & & & x_3 & & \\ & x_{10} & & & x_{11} & & & x_{12} & \\ & & x_{19} & & & x_{20} & & & x_{21} \\ \hline x_4 & & & x_5 & & & x_6 & & \\ & x_{13} & & & x_{14} & & & x_{15} & \\ & & x_{22} & & & x_{23} & & & x_{24} \\ \hline x_7 & & & x_8 & & & x_9 & & \\ & x_{16} & & & x_{17} & & & x_{18} & \\ & & x_{25} & & & x_{26} & & & x_{27} \end{array}\right].\end{aligned}$$

Then, it follows immediately that $\tilde{\tilde{\mathbf{K}}}_{block} = \tilde{\tilde{\mathbf{K}}}_{diag}$:

$$\tilde{\tilde{\mathbf{K}}}_{block} = \tilde{\tilde{\mathbf{K}}}_{diag} = \left[\begin{array}{ccc|ccc|ccc} x_1 & & & & & & & & \\ & x_{10} & & & & & & & \\ & & x_{19} & & & & & & \\ \hline & & & x_5 & & & & & \\ & & & & x_{14} & & & & \\ & & & & & x_{23} & & & \\ \hline & & & & & & x_9 & & \\ & & & & & & & x_{18} & \\ & & & & & & & & x_{27} \end{array}\right].$$

□

Appendix F

Plots of the Original and SoV Wavenumber

In this appendix, we give and compare the plots of the original and the SoV wavenumber, for the wedge, sinus, random and min-max model, respectively, in the case of $M, N = 35$. Note that, for the constant the and rectangular model, the original and SoV wavenumber are *identical* and therefore they are not treated in this appendix.

Recall that we have computed matrix $\mathbf{K}$, which can be decomposed into:

$$\mathbf{K} = \mathbf{I}_y \otimes \mathbf{K}_x + \mathbf{K}_y \otimes \mathbf{I}_x + \widetilde{\mathbf{K}}. \tag{F.1}$$

In the SoV preconditioner, we assume $\widetilde{\mathbf{K}} = 0$. This gives

$$\hat{\mathbf{K}} = \mathbf{I}_y \otimes \mathbf{K}_x + \mathbf{K}_y \otimes \mathbf{I}_x. \tag{F.2}$$

First, matrix $\hat{\mathbf{K}}$ is given in the left subplots and, second, matrix $\hat{\mathbf{K}}$ is given in the right subplots and is used as wavenumber in the SoV preconditioning.

Notice that there are relatively large differences between k_{SoV} and k in all models, considering all figures in this appendix.

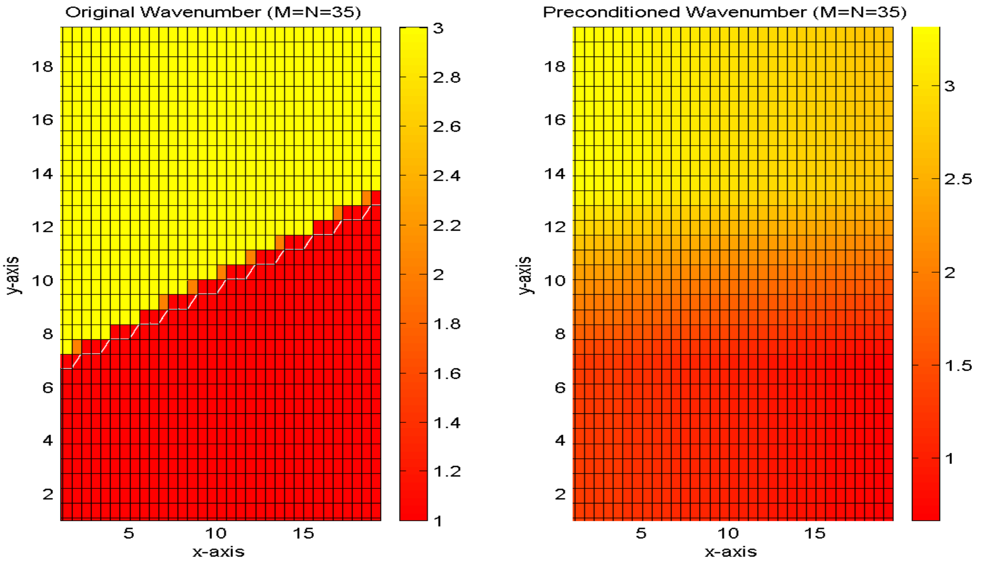


Figure F.1: Wavenumber k in the wedge model. Left: original wavenumber. Right: wavenumber in the preconditioned case.

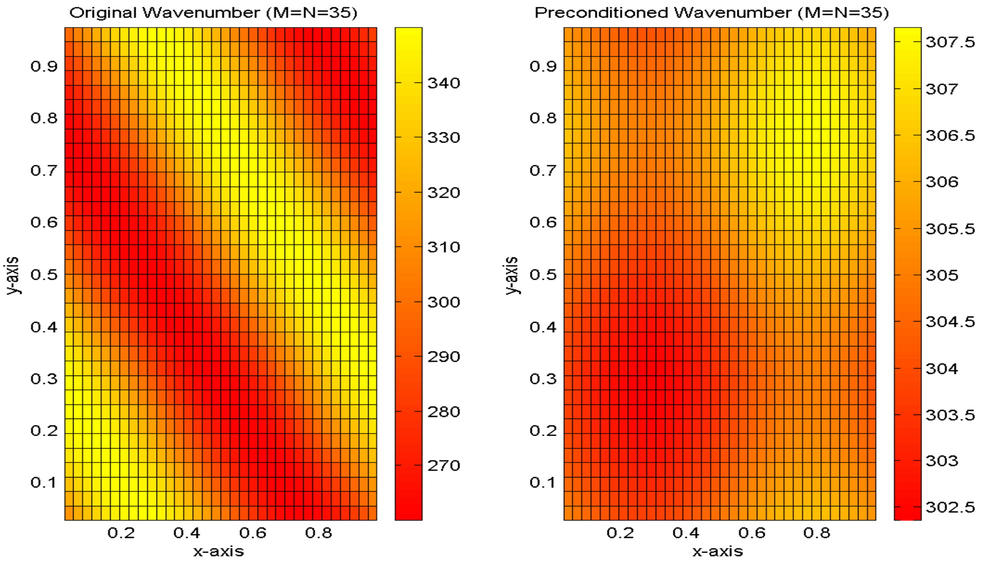


Figure F.2: Ssinus model with wavenumber k in the sinus model. Left: original wavenumber. Right: wavenumber in the preconditioned case.

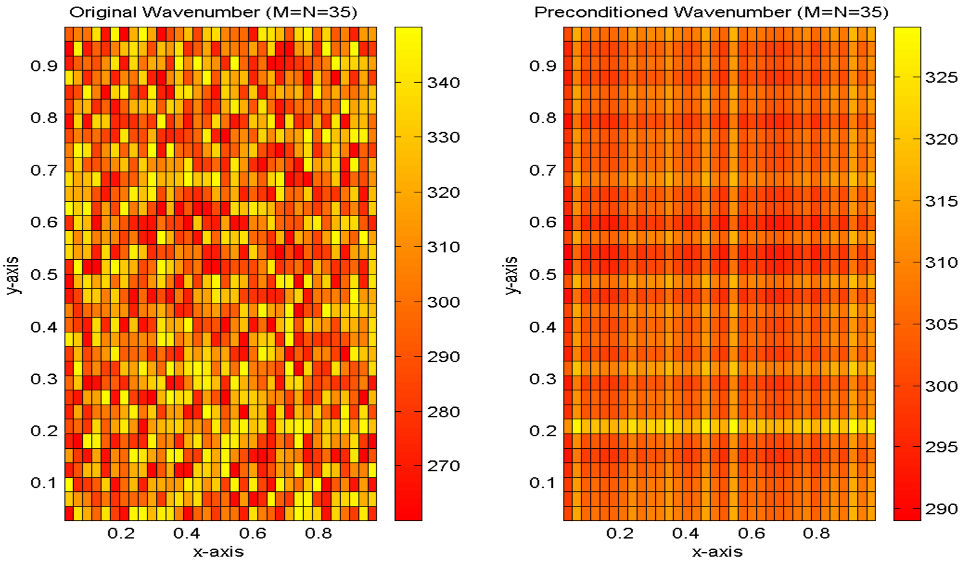


Figure F.3: Random model with wavenumber k in the random model. Left: original wavenumber. Right: wavenumber in the preconditioned case.

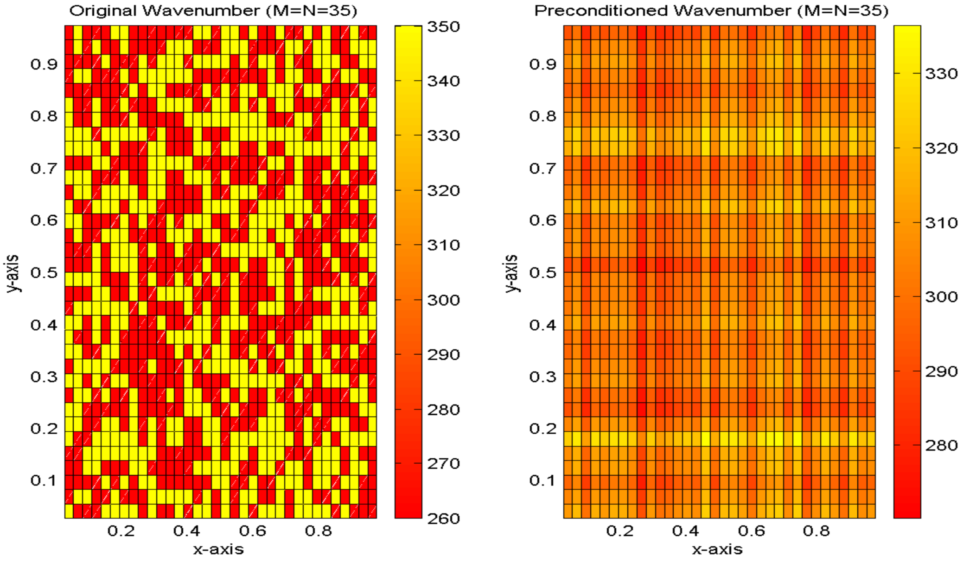
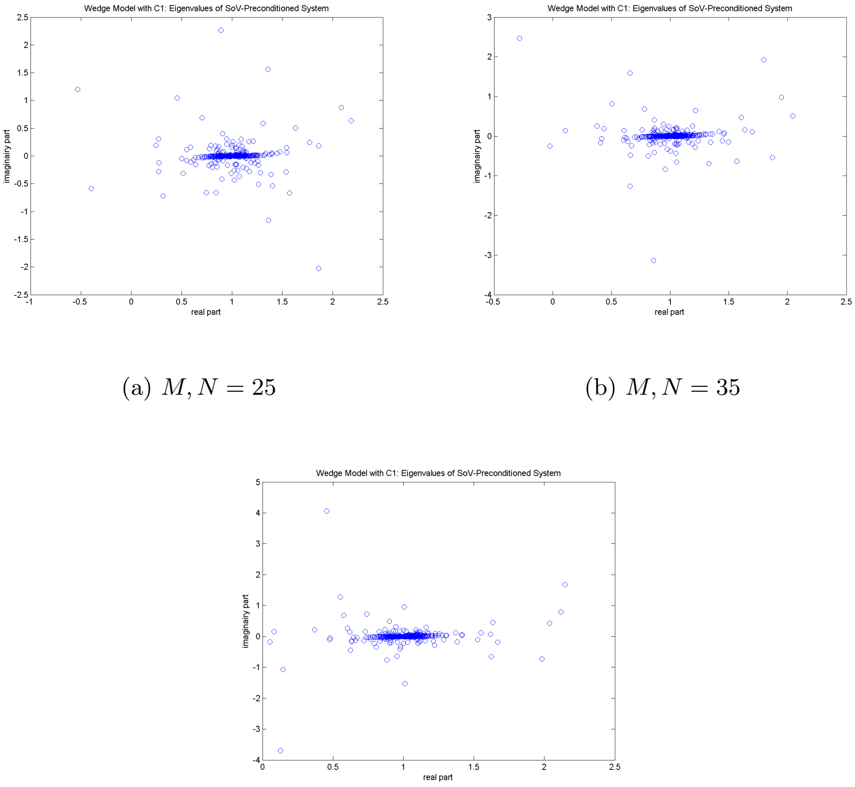


Figure F.4: Plot of the min-max model with wavenumber k in the min-max model. Left: original wavenumber. Right: wavenumber in the preconditioned case.

Appendix G

Eigenvalue Plots

Some plots of the eigenvalues of SoV preconditioned systems with alternative $k_x(x)$ and $k_y(y)$ are given in this appendix.

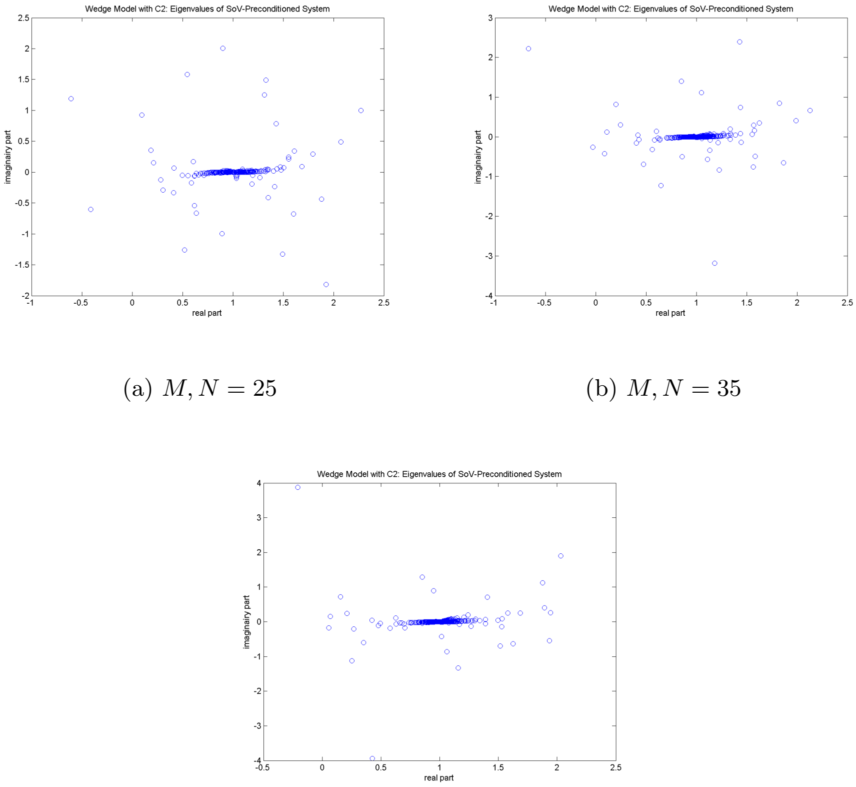


(a) $M, N = 25$

(b) $M, N = 35$

(c) $M, N = 45$

Figure G.1: Eigenvalues of the system $\boldsymbol{M_{SoV}^{-1}A}$ ***with*** $M, N = 25, 35, 45$ ***using choice (C1), where test problem (V) is used in combination with the wedge model.***



(a) $M, N = 25$

(b) $M, N = 35$

(c) $M, N = 45$

Figure G.2: Eigenvalues of the system $M_{SoV}^{-1}A$ ***with*** $M, N = 25, 35, 45$ ***using choice (C2), where test problem (V) is used in combination with the wedge model.***

Appendix H

Errors using $\mathbf{W}_L = \mathbf{W}_R$ in SoV

Since matrix $\widetilde{\mathbf{W}}_L^H$ is only an approximation of the left eigenvector matrix $\mathbf{W}_L^H$, some errors are made during the iterative process. Although this approximation has more or less no influence on the convergence (see Table 7.3), it makes sense to investigate this observation further.

If we compute

$$(\mathbf{I}_y \otimes \mathbf{W}_R)(\mathbf{I}_y \otimes \mathbf{W}_L^H) = \mathbf{I}_y \otimes (\mathbf{W}_R \mathbf{W}_L^H), \tag{H.1}$$

then this leads exactly to the identity matrix. However, in our case using $\widetilde{\mathbf{W}}_L^H = \mathbf{W}_R^H$, it results in some 'redundant' terms at the places of the zeros for sufficient large M and N. To investigate this, we define the following norm

$$||\mathbf{I}_y \otimes (\mathbf{W}_R \mathbf{W}_L^H) - \mathbf{I}||_2, \tag{H.2}$$

where $\mathbf{I}$ is the unit matrix of length MN. The results of some experiments can be found in Table H.1.

M, N	$\|\|\mathbf{I}_y \otimes (\mathbf{W}_R \widetilde{\mathbf{W}}_L^H) - \mathbf{I}\|\|_2$	$\|\|\mathbf{I}_y \otimes (\mathbf{W}_R \mathbf{W}_L^H) - \mathbf{I}\|\|_2$
5	0.09	1.55×10^{-16}
15	0.68	6.67×10^{-16}
25	1.08	8.46×10^{-16}
35	1.30	2.19×10^{-15}
45	1.52	4.06×10^{-15}

Table H.1: Norm of the matrix of differences between two unit matrices in our test problem (V) using the wedge model.

In Table H.1, one concludes that using $\widetilde{\mathbf{W}}_L^H$ can clearly lead to significant errors, especially for large M and N, while the errors using $\mathbf{W}_R^{-1}$ are equal to zero.

The errors can also be seen in the following tests. We apply the *original* $\widetilde{\mathbf{K}}$ in the preconditioner (leading to a non-diagonal-block system) in two ways:

Test 1 Apply $\widetilde{\mathbf{K}}$ as defined in the decomposition $\tilde{k}^2(x,y) = k^2(x,y) - k_x^2(x) - k_y^2(y)$. In general, this converges very fast, but not in one iteration due to the approximated boundary conditions;

Test 2 Apply $\widetilde{\mathbf{K}}$ as defined in Test 1, but now with an extra computation:

$$\widetilde{\widetilde{\mathbf{K}}} = \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H)\ \widetilde{\mathbf{K}}\ (\mathbf{I}_y \otimes \mathbf{W}_R)\mathbf{P}, \tag{H.3}$$

followed by

$$(\mathbf{I}_y \otimes \mathbf{W}_R)\mathbf{P}\ \widetilde{\widetilde{\mathbf{K}}}\ \mathbf{P}^T(\mathbf{I}_y \otimes \mathbf{W}_L^H). \tag{H.4}$$

Theoretically, after the computations, this has to result in the matrix $\widetilde{\mathbf{K}}$.

The results of these tests can be found in Table H.2.

	$\widetilde{\mathbf{W}}_L^H = \mathbf{W}_R^H$		$\mathbf{W}_L^H = \mathbf{W}_R^{-1}$	
M, N	Test 1	Test 2	Test 1	Test 2
15	5	17	5	5
25	3	12	6	6

Table H.2: Number of iterations of Bi-CGSTAB with preconditioner of Test 1 and Test 2 using both choices of $\mathbf{W}_L^H$ in test problem (V) in combination with the wedge model.

One can see that the case with $\widetilde{\mathbf{W}}_L^H$ gives *poor* results. However, to derive $\widetilde{\mathbf{K}}$ as in Expression (H.4), we do *not* need the entire matrix $\widetilde{\widetilde{\mathbf{K}}}$. Only the main block-diagonals are required. Therefore the differences between using $\widetilde{\mathbf{W}}_L^H$ or $\widetilde{\mathbf{W}}_L^H$ are small, see Table 7.3.

However, in the remaining of this thesis we apply only $\mathbf{W}_L^H = \mathbf{W}_R^{-1}$ to be absolutely sure of the correctness of the left eigenvector matrix.

Appendix I

Including a Block Diagonal $\tilde{\tilde{\mathbf{K}}}$ in SoV

In Section 7.4, we have seen that an addition of diagonal matrix $\tilde{\tilde{\mathbf{K}}}_{mod}$ to the SoV preconditioner leads to better convergence results for iterative methods. In this appendix, we examine the possibilities to choose a *block*-diagonal matrix $\tilde{\tilde{\mathbf{K}}}_B$, such that the SoV preconditioner including this matrix is more efficient in iterative methods. At first, we look at the eigenvalues of the system $\tilde{\tilde{\mathbf{K}}}_B^{-1}\tilde{\tilde{\mathbf{K}}}$. The criterium is to choose α_i (see the next section for the definition of α_i) such that all eigenvalues are as far as possible from zero. In future research, other more appropriate criteria can be chosen.

I.1 Example

Consider a HP with sizes $M \times N = 2 \times 2$ and assume that

$$\tilde{\tilde{\mathbf{K}}} = \left[\begin{array}{cc|cc} 2 & & 1 & \\ & 2 & & 1 \\ \hline 1 & & 2 & \\ & 1 & & 2 \end{array}\right]. \tag{I.1}$$

In Section 7.4, we have used a diagonal matrix $\tilde{\tilde{\mathbf{K}}}_{mod}$ as additional term in the preconditioner:

$$\tilde{\tilde{\mathbf{K}}}_{mod} = \left[\begin{array}{cc|cc} 2 & & & \\ & 2 & & \\ \hline & & 2 & \\ & & & 2 \end{array}\right]. \tag{I.2}$$

Now, can we choose a block diagonal matrix $\tilde{\tilde{\mathbf{K}}}_B$ of the form

$$\tilde{\tilde{\mathbf{K}}}_B = \left[\begin{array}{cc|cc} 2 & \alpha_3 & & \\ \alpha_1 & 2 & & \\ \hline & & 2 & \alpha_4 \\ & & \alpha_2 & 2 \end{array}\right], \tag{I.3}$$

which speeds up the convergence of the iterative method?

I.1.1 Identical α_i

Assume that $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 \in \mathbb{R}$ which we denote by α. It appears that in this situation $\alpha = 0$ is the best choice. This can also be seen in Figure I.1, which has been made in MATLAB with the help of a code that computes the minimum eigenvalues of $\overset{\approx}{\mathbf{K}}_B^{-1}\overset{\approx}{\mathbf{K}}$, by varying α in the region $[-1.5, 1.5]$ with stepsize $h = 0.1$. [1]

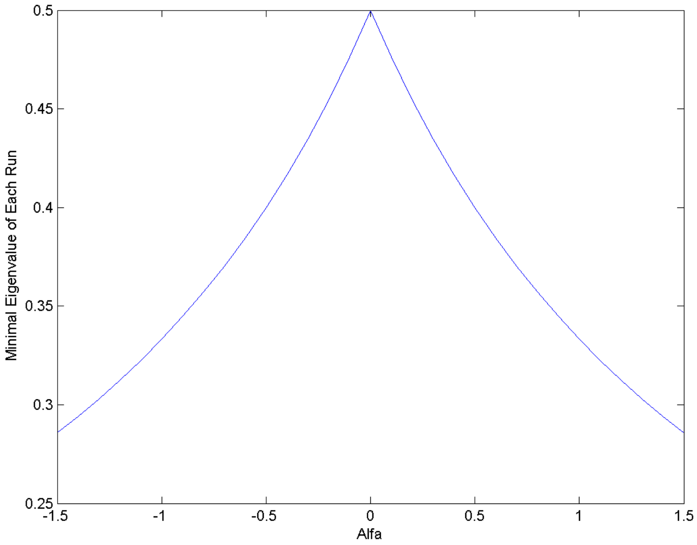


Figure I.1: The minimum eigenvalue of each run with different value of α.

At $\alpha = 0$, the maximum value of the plot is reached. We see that the minimum eigenvalue is $\lambda_{\text{min}} = 0.5$ in this case.

It appears in tests that choosing another values at the positions with value 1 in the matrix, as defined in (I.1), leads to the same result: $\alpha = 0$ is optimal.

I.1.2 Different α_i

In the tests, we can show that in cases of different α_i leads to another results than those of the previous subsection. For instance: the choices

$$\left\{\begin{array}{lcr} \alpha_1 & = & -1.5; \\ \alpha_2 & = & 1.5; \\ \alpha_3 & = & 1.0; \\ \alpha_4 & = & -1.0, \end{array}\right. \tag{I.4}$$

leads to $\mathcal{R}(\lambda_{\text{min}}) = 0.73$. We have used $\alpha_i \in [-1.5, 1.5]$ with stepsize $h = 0.25$ to limit the computational work.

[1] It appears that taking larger intervals than $[-1.5, 1.5]$ leads to *singular* matrices $\overset{\approx}{\mathbf{K}}_B^{-1}\overset{\approx}{\mathbf{K}}$.

I.1.3 Different α_i in Blocks

It may be inefficient in the SoV-preconditioner to use fully different values of α_i, as considered in the previous subsection. Instead of using these, we apply identical blocks in expression (I.3), leading to

$$\tilde{\tilde{\mathbf{K}}}_B = \left[\begin{array}{cc|cc} 2 & \alpha_2 & & \\ \alpha_1 & 2 & & \\ \hline & & 2 & \alpha_2 \\ & & \alpha_1 & 2 \end{array}\right]. \tag{I.5}$$

It appears that in this case $\alpha_1 = \alpha_2 = 0$ is optimal, leading to $\lambda_{\text{min}} = 0.5$. We make the observation that this choice is not unique; for instance $\alpha_1 = \alpha_2 = -0.2$ or $\alpha_1 = \alpha_2 = -1.35$ lead also to $\mathcal{R}(\lambda_{\text{min}}) = 0.5$.

I.2 Conclusion and Outlook

Considering the previous section, one can conclude that it may be possible to find values of $\alpha_i \neq 0$, which leads to a better eigenvalue distribution of $\tilde{\tilde{\mathbf{K}}}_B^{-1} \tilde{\tilde{\mathbf{K}}}$, in the case of the possibility of choosing all α_i differently. This may result in better performance of the full SoV preconditioner.

In future, we may examine

- whether it is indeed inefficient to choose different diagonal blocks (and therefore different α_i) in expression (I.3);
- how to choose α_i properly for each i in our 2-D testproblems of the wedge model;
- how large is the influence of $\tilde{\tilde{\mathbf{K}}}_B$ in the preconditioner.

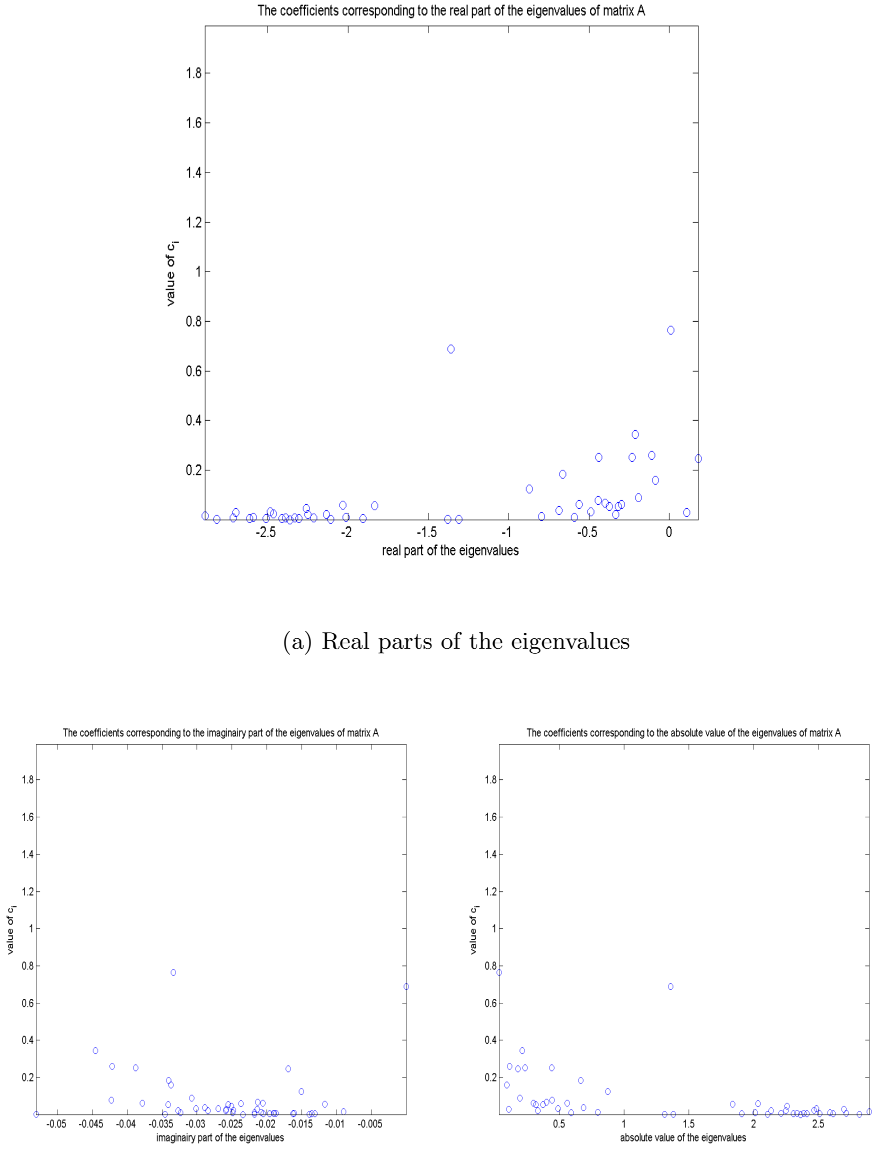
Appendix J

Second Extension to Example 2

In Figure J.1, one can find the results of Example 2: the weight of the coefficients with respect to the eigenvalues.

Comparing to the extension of Example 1 (see Subsection 8.3.1), we have only a few eigenvalues, but it can also be seen in these figures that the absolute and real parts of the eigenvalues around zero are important. Note that there is exactly one 'lost' eigenvalue with coordinates $(-1.35, 0.65)$ that also has a large influence on the solution.

Next, we compare the real and imaginary parts and absolute values of the coefficients. The results for Example 2 can be found in Figure J.2. We found the same kind of results as in Example 1.



(a) Real parts of the eigenvalues

(b) imaginary parts of the eigenvalues

(c) Absolute values of the eigenvalues

Figure J.1: Real parts of the coefficients corresponding to the real, imaginary and absolute parts of the eigenvalues of A, respectively, in the case of $M, N = 7$***.***

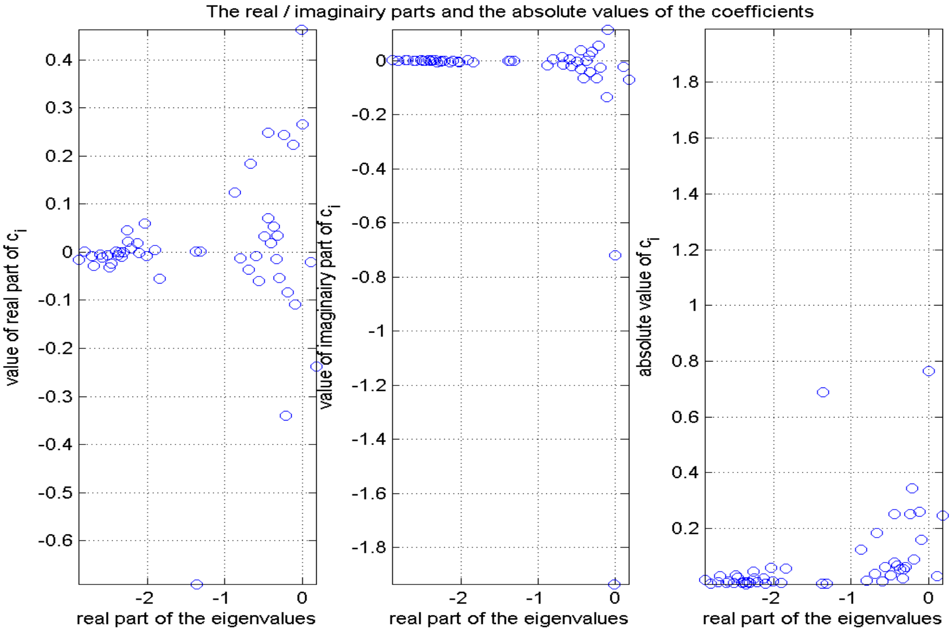


Figure J.2: Real part, imaginary part and the absolute values of the coefficients corresponding to the real parts of the eigenvalues of A in the case of $M, N = 7$.

Appendix K

Successive Refinement for the 1-dimensional HP

Considering the eigenvalue distributions of the SoV preconditioned systems, there are possibilities to apply *successive refinement* techniques to solve these systems.

We start with the *1-dimensional* Helmholtz problem (HP). The SoV preconditioner is exact in this case (i.e., the iterative method needs only one iteration to converge). In this appendix, we apply the CSL preconditioner in the Bi-CGSTAB method and we use the successive refinement technique to solve the HP.

K.1 Problem Formulation

The one-dimensional HP is defined by

$$-\frac{\mathrm{d}^2}{\mathrm{d}x^2}p(x) - k(x)^2 p(x) = f(x), \quad x \in \Omega, \tag{K.1}$$

where we apply Dirichlet conditions for simplicity:

$$p(x) = 0, \quad x \in \partial\Omega. \tag{K.2}$$

We take the unit domain $\Omega = (0,1)$. Moreover,

$$k^2(x) = \begin{cases} k_1^2 & \text{if } x < 1/2; \\ \frac{k_1^2+k_2^2}{2} & \text{if } x = 1/2; \\ k_2^2 & \text{if } x > 1/2, \end{cases} \tag{K.3}$$

and furthermore, we take the source term $f(x)$ to be constant:

$$f(x) = 1 \quad \forall x. \tag{K.4}$$

After numerical discretization and second-order finite differences we obtain the linear system:

$$\mathbf{A}\mathbf{p} = \mathbf{f}. \tag{K.5}$$

K.2 Successive Refinement Technique

The resulting linear system (K.5) is solved with Bi-CGSTAB using an appropriate starting vector. We find a 'good' starting vector such that Bi-CGSTAB requires less iterations to converge. This will be done with *successive refinement.*

The idea of successive refinement is that, if the original grid is of size N, then we start with finding the solution $\mathbf{q}$ on a grid of size $\frac{1}{2}(N-1)$ and, thereafter, we prolongate this solution to $\hat{\mathbf{p}}$ which will be used as starting vector $\mathbf{p}_0$ for Bi-CGSTAB. The algorithm of the *prolongation* is given below.

Algorithm K.1: Prolongation

```
for $i = 2$ to $N-1$
    if $\mathrm{mod}(i,2) = 0$
        $\hat{p}_i = q_{i/2}$
    else
        $\hat{p}_i = \frac{1}{2}\left(q_{(i-1)/2} + q_{(i+1)/2}\right)$
    endif
    $\hat{p}_1 = 0.5 \cdot q_1$
    $\hat{p}_N = 0.5 \cdot q_{(N-1)/2}$
endfor
```

One is referred to Wesseling [54] for a more mathematical treatment of prolongation methods.

For example, in the case of $N = 45$, we first start with $N = 21$ and find the solution $\mathbf{q}$ for this simpler problem. Then, we compute the prolongated solution $\hat{\mathbf{p}}$ of length $N = 45$, with the help of the linear prolongation formulae as described in Algorithm K.1.

K.2.1 Gauss-Seidel Iterations

It appears in our numerical experiments that $||\mathbf{p} - \mathbf{p}_0||_2$ can be very small, whereas the residual $||\mathbf{f} - \mathbf{A}\mathbf{p}_0||_2$ remains relatively large.

To remedy the relatively large residuals, we apply a few Gauss-Seidel (GS) iterations with $\mathbf{p}_0$. This 'smooths' the starting vector such that the residual will be smaller [1]. The resulting vector $\hat{\mathbf{p}}_0$ will be used as the new starting vector of Bi-CGSTAB. In fact, we follow the following steps in the whole procedure:

1. solve $\mathbf{q}$ on the coarse grid;
2. compute $\hat{\mathbf{p}}$ after prolongation of $\mathbf{q}$;
3. compute $\hat{\mathbf{p}}_0$ with a few Gauss-Seidel iterations;
4. use $\hat{\mathbf{p}}_0$ as starting vector in Bi-CGSTAB.

In the test problems of this appendix, we use 2 and 10 GS iterations, respectively, denoting by 'GS2' and 'GS10'.

[1] In future, we can also apply Gauss-Seidel to the gridpoints which are not points at the coarse grid. This may lead to smaller relative residuals.

K.3 Parameters in the HP

Since the iterative method in combination with CSL converges very fast, we have to increase the wavenumbers k_1^2 and k_2^2, to obtain a method with sufficient iterations to analyze. Moreover, we have to also increase N to keep an accurate solution. Recall the following relations, as given in Chapter 2:

$$\left\{\begin{array}{rcl} k_{\text{max}} & = & \dfrac{2\pi}{\lambda_{\text{min}}}; \\[2ex] \lambda_{\text{min}} & = & \dfrac{L}{W_{\text{max}}}; \\[2ex] W_{\text{max}} & = & \dfrac{N}{G}, \end{array}\right. \tag{K.6}$$

where W_{max} is the maximum number of waves in Ω, G is the minimum number of gridpoints per wavelength, N is the number of gridelements, L is the length of each direction of Ω, λ_{min} is the minimum wavelength and, finally, k_{max} is the maximum wavenumber in the problem. With the help of the expressions in (K.6), we obtain the following relation of N and k_{max}^2:

$$N = \sqrt{\frac{G^2L^2}{4\pi^2}k_{\text{max}}^2}. \tag{K.7}$$

Taking $G = 15$ and $L = 1$, we obtain:

$$N = \sqrt{\frac{225}{4\pi^2}k_{\text{max}}^2} \approx 2.4\sqrt{k_{\text{max}}^2}. \tag{K.8}$$

Moreover, since we have $k_1^2 \leq k_2^2$ in our test runs , it yields: $k_{\text{max}}^2 = k_2^2$. This leads to

$$N \approx 2.4\sqrt{k_2^2}. \tag{K.9}$$

For example, if we take $k_1^2 = 1000, k_2^2 = 2000$, then at least

$$N \approx 2.4\sqrt{2000} = 107 \text{ gridpoints} \tag{K.10}$$

are required to be ensured of an accurate solution.

In our test runs, it appears that our computer is only able to compute with at most about $N = 2000$. Hence, the maximum k_2^2 is

$$k_{\text{max}}^2 = \frac{N^2}{2.4^2} = 6.94 \cdot 10^5. \tag{K.11}$$

K.4 Results and Analysis

The results of the test problem with varying wavenumbers k_1 and k_2, using successive refinement and a few Gauss Seidel steps, can be found in Table K.1.

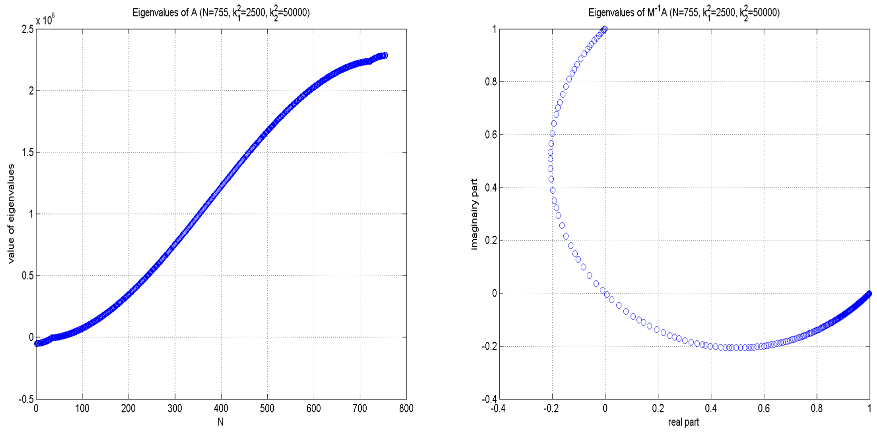
In Table K.1, we can see that using a starting vector and using smoothing Gauss Seidel steps, lead to rather good results, especially for relatively large problems. In row 3 of this table, we observe that the convergence of the iterative method is more than twice as fast in that case!

Furthermore, the eigenvalue distributions of both matrix $\mathbf{A}$ and matrix $\mathbf{M}^{-1}\mathbf{A}$ of a test run have been plotted in Figures K.1(a) and K.1(b). One observes immediately that the eigenvalues of $\mathbf{M}^{-1}\mathbf{A}$ are complex. Moreover, the range of the spectra $\mathbf{M}^{-1}\mathbf{A}$ is obviously smaller than the range of the spectra of $\mathbf{A}$, which *may* be favorable in the successive refinement method.

N	k_1^2	k_2^2	It.	It.$(\mathbf{p}_0)$	It.$(\mathbf{p}_{0,GS2})$	It.$(\mathbf{p}_{0,GS10})$
755	2.5×10^3	5.0×10^4	120	125	124	122
955	5.0×10^3	1.0×10^5	243	184	179	171
1855	1.0×10^4	5.0×10^5	2994	1255	648	603
1855	1.0×10^5	5.0×10^5	667	531	516	492

N	$\|\|\mathbf{f} - \mathbf{A}\mathbf{p}_0\|\|_2$	$\|\|\mathbf{f} - \mathbf{A}\mathbf{p}_{0,GS2}\|\|_2$	$\|\|\mathbf{f} - \mathbf{A}\mathbf{p}_{0,GS10}\|\|_2$
755	19.6	2.3	0.32
955	219.3	27.6	6.2
1855	117.5	15.5	4.2
1855	2027.3	267	73.9

Table K.1: Comparison of Bi-CGSTAB in combination with the CSL preconditioner. First, without and, secondly, with starting vector $\mathbf{p}_0$ obtained with successive refinement and also 2 and 10 Gauss Seidel iterations (GS2 and GS10, respectively). The norms of residuals are also given and the number of iterations is denoted by 'It.'.



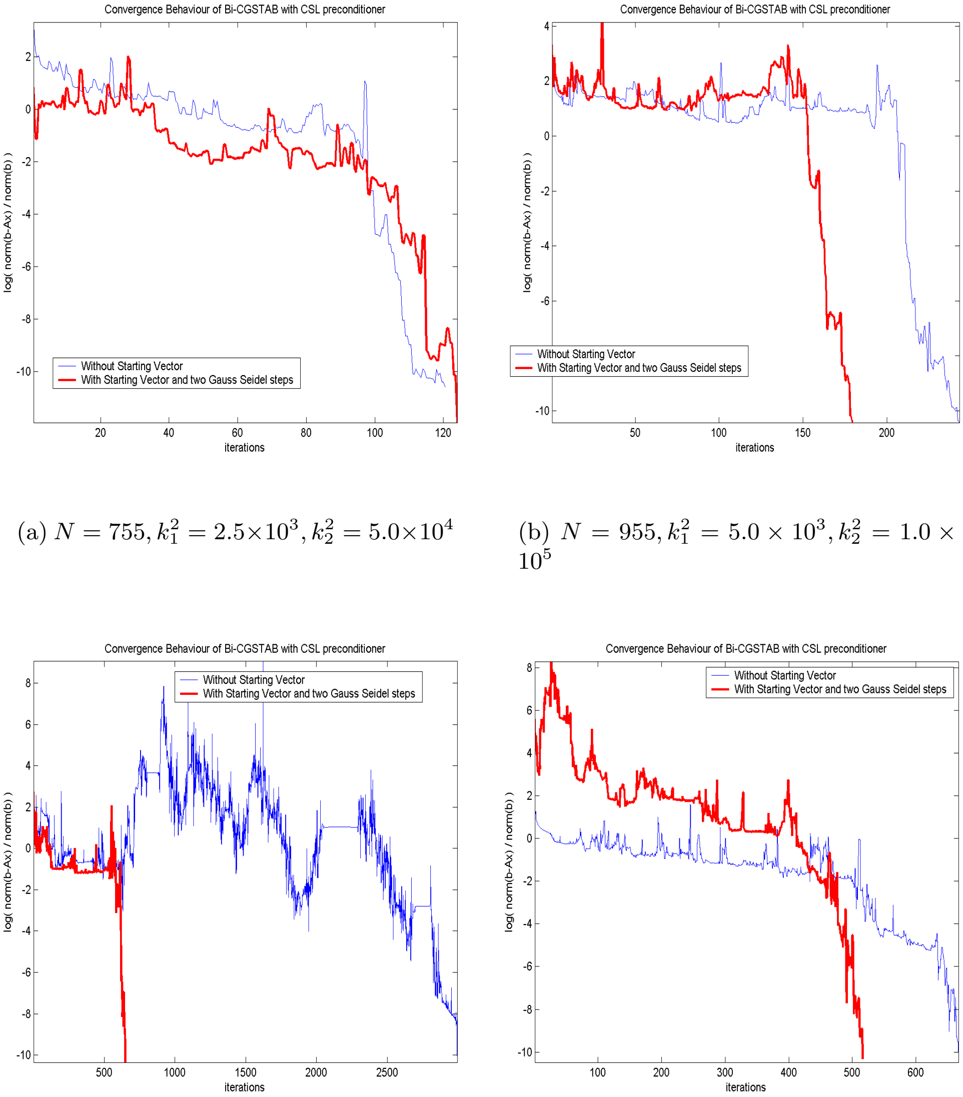
(a) matrix $\mathbf{A}$

(b) matrix $\mathbf{M}^{-1}\mathbf{A}$

Figure K.1: Eigenvalues of the original and the preconditioned system with $N = 755, k_1^2 = 2.5 \times 10^3$ and $k_2^2 = 5.0 \times 10^4$.

Finally, the logarithms of the relative residuals of the test runs, used in

Table K.1, are given in Figure K.2.



(a) $N = 755, k_1^2 = 2.5{\times}10^3, k_2^2 = 5.0{\times}10^4$

(b) $N = 955, k_1^2 = 5.0 \times 10^3, k_2^2 = 1.0 \times 10^5$

(c) $N = 1855, k_1^2 = 1.0 \times 10^4, k_2^2 = 5.0 \times 10^5$

(d) $N = 1855, k_1^2 = 1.0 \times 10^5, k_2^2 = 5.0 \times 10^5$

Figure K.2:** **Logarithms of the relative residuals during the iterations with Bi-CGSTAB in combination with CSL without starting vector and with $\hat{p}_0$ including two Gauss Seidel steps, respectively.

K.4.1 Conclusions and Future Research

We have seen that the successive refinement technique, in combination with the Gauss-Seidel method, makes sense for the CSL preconditioned system.

In Figure K.2, we see the strongly erratic behavior of the residuals using Bi-

CGSTAB. In future, we could apply GMRES instead of Bi-CGSTAB to avoid the erratic behavior, which may easier to analyze the improvements using the successive refinement technique.

Moreover, in future research, the 2-dimensional Helmholtz problem could be implemented, whereafter we can investigate the SoV preconditioner, since this is not exact anymore. The same procedure of the successive refinement technique can be followed as in the 1-dimensional case.