Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

Iterative solutions to sequences of Helmholtz equations

A thesis submitted to the
Delft Institute of Applied Mathematics
in partial fulfillment of the requirements

for the degree

MASTER OF SCIENCE
in
APPLIED MATHEMATICS

by

J.M. DE GIER

Delft, the Netherlands
August 2012

TU Delft

MSc THESIS APPLIED MATHEMATICS

"Iterative solutions to sequences of Helmholtz equations"

J.M. DE GIER

Delft University of Technology

Daily supervisor

Dr.ir. M.B. van Gijzen

Responsible professor

Prof.dr.ir. C. Vuik

Other thesis committee members

Dr. K. Dekker

Dr. J.L.A. Dubbeldam

August 2012

Delft, the Netherlands

Preface

The so-called Pythagoreans, who were the first to take up mathematics, not only advanced this subject, but saturated with it, they fancied that the principles of mathematics were the principles of all things.
Aristotle

For the last 300 or so years, the exact sciences have been dominated by what is really a good idea, which is the idea that one can describe the natural world using mathematical equations.
Stephen Wolfram

During my studies in Applied Mathematics I frequently wondered why mathematics is as effective as it is. If mathematics is just a human construction that consists of symbols and relations, how can it match the reality that surrounds us to amazing precision? It seems that the mathematical building blocks correspond to the building blocks of the universe and this is one of the reason why I love my studies in mathematics: it is a way to comprehend the (physical) reality.

With this thesis I finish my education in Applied Mathematics at the Delft University of Technology. In the past year, I gradually changed from a student to a 'citizen', as it is called among students. This means less late-night partying, working 9-to-5 and, as icing on the cake, moving from a student house to an apartment.

I would like to thank the thesis committee members for their time and energy spent on reading the literature review and this thesis and their comments to improve the research and the thesis. My biggest thanks go to my supervisor, Martin van Gijzen, for the many conversations that guided my research and clarified my thinking on any problem that I came across.

Jan de Gier
Delft, the Netherlands, August 2012

List of Symbols

Mathematical symbols

i	imaginary unit, $\sqrt{-1}$
$\overline{\alpha+\beta i}$	complex conjugate, $\alpha-\beta i$
$\lvert\alpha+\beta i\rvert$	absolute value, $\sqrt{\alpha^2+\beta^2}$
$\varphi_{\alpha+\beta i}$	argument, $\arctan(\beta/\alpha)$
$\mathbf{x}$	vector $(x_1,\ldots,x_n)^T$ in $\mathbb{R}^n$ or $\mathbb{C}^n$ or vector field $S\to\mathbb{R}^n$ or $S\to\mathbb{C}^n$
$\mathcal{K}_k(\mathbf{A},\mathbf{r}_0)$	Krylov subspace of dimension k, $\operatorname{span}\{\mathbf{r}_0,\mathbf{A}\mathbf{r}_0,\ldots,\mathbf{A}^{k-1}\mathbf{r}_0\}$
$\mathbf{x}\cdot\mathbf{y}$	$\sum_{i=1}^{n}\overline{x}_i y_i$
$\lVert\mathbf{x}\rVert$	2-norm, $\sqrt{\mathbf{x}\cdot\mathbf{x}}$
$\Delta\mathbf{x}_i$	$\mathbf{x}_i-\mathbf{x}_{i-1}$
$\mathbf{A}$	$n\times n$ real or complex matrix, $(a_{ij})^{n\times n}$ in $\mathbb{R}^{n\times n}$ or $\mathbb{C}^{n\times n}$
$\underline{\mathbf{A}}$	$(n+1)\times n$ real or complex matrix, the extended counterpart of $\mathbf{A}$
$\mathbf{A}^*$	conjugate transpose of $\mathbf{A}$, $\overline{\mathbf{A}^T}$
$\sigma(\mathbf{A})$	spectrum of $\mathbf{A}$, set of eigenvalues of $\mathbf{A}$
$\rho(\mathbf{A})$	spectral radius of $\mathbf{A}$, maximum of the absolute values of the set of eigenvalues of $\mathbf{A}$
$\kappa(\mathbf{A})$	spectral condition number of (non-singular) $\mathbf{A}$ in the 2-norm, $\lVert\mathbf{A}\rVert\lVert\mathbf{A}^{-1}\rVert$
Ω	bounded region in $\mathbb{R}^n$
Γ	boundary of Ω
$\mathbf{n}$	outward unit normal to the boundary
$\dfrac{\partial f}{\partial x_i}$	partial derivative of f with respect to x_i
∇f	gradient of f, $\left(\dfrac{\partial f}{\partial x_1},\ldots,\dfrac{\partial f}{\partial x_n}\right)^T$
$\dfrac{\partial f}{\partial n}$	$\nabla f\cdot\mathbf{n}$
$\nabla\cdot\mathbf{f}$	divergence of $\mathbf{f}$, $\sum_{i=1}^{n}\dfrac{\partial f_i}{\partial x_i}$

$\nabla^2 f$	Laplacian of f, $\sum_{i=1}^{n} \frac{\partial^2 f}{\partial x_i^2}$, $\nabla \cdot \nabla f$
$\delta(\mathbf{x})$	n-dimensional Dirac delta function
δ_{ij}	Kronecker delta

Physical symbols

t	time [s]
ρ	(fluid) density [kg/m^3]
p	(fluid) pressure [kg/(m s^2)]
$\mathbf{u}$	(particle) displacement [m]
$\mathbf{v}$	(particle) velocity [m/s]
c_0	speed of sound (in air) [m/s]
f	(ordinary) frequency [1/s]
ω	angular frequency [1/s]
k	acoustic wavenumber [1/m]
λ	wavelength [m]
Z_0	characteristic acoustic impedance (of air) [kg/(s m^2)]
Z_n	specific normal acoustic impedance [kg/(s m^2)]

List of Figures

Page

List of Tables

Page

Contents

Chapter 1

Introduction

The essence of mathematics is not to make simple things complicated,
but to make complicated things simple.
Stanley Gudder

Background noise in cars is discomforting for occupants and hence the reduction of the noise emissions is considered to be an important aspect in the design of new cars. In order to reduce the noise in a car without the need of an expensive prototype, a mathematical model of the acoustic behaviour of this car and the propagation of sound inside this car is needed. At the basis of this model lies a system of coupled differential equations that describes for instance the sources of noise, such as vibrating car parts due to road contact and the engine, and the propagation of this noise through the structure of and the air inside the car. Since different sound frequencies result in different ways of propagation, the differential equations need to be solved for a whole range of frequencies. The combined results for all relevant frequencies might lead to adjustments in the car design, after which the acoustics of the adjusted car are again modelled and the propagation of sound with certain frequencies is recomputed. Although much research is carried out in for instance the development of numerical models and numerical methods for the simulation and optimisation of the acoustic models, it is not yet possible to minimise the noise inside a car based on the model solutions.

In this thesis, we consider sequences of linear systems of equations that follow from the differential equations that describe the acoustics of a car, where the solution to a single linear system corresponds to the pressure perturbations that are caused by acoustic waves of a certain frequency. We consider several iterative numerical methods which belong to the class of Krylov subspace methods and that solve these linear systems. In addition we investigate a particular type of so-called preconditioning, which is a specific procedure that rewrites the system into a form that is more suitable for the numerical methods.

Since we need to solve the acoustic problem for sequences of frequencies, there is an order in which all individual problems are solved. This offers opportunities in the use of the results that are already obtained and we study a number of these options in this thesis. The research question reads: how can information from previous linear systems be used to reduce the computation time needed? This information considers solution vectors, spectral information of the system matrices and results on the numbers of iterations that are needed for convergence. We can use this information for initial search spaces, for improving the initial guess and for adapting the preconditioner. We consider the research question in the context of IDR(s) and the preconditioners we consider are (based on) the shifted Laplace preconditioners.

1.1 Content of the report

This thesis comprises eight chapters. In the second chapter we derive the wave equation, which describes the pressure perturbations in a fluid that are caused by acoustic waves. We simplify it

to the Helmholtz equation, which is a time independent equivalent of this wave equation. Chapter three briefly describes the discretisation of the Helmholtz equation that results in a linear system of equations and introduces an academic Helmholtz problem that is used for the testing and prototyping that can be found in Appendix B. Chapter four treats several numerical methods and general solution techniques that are applicable to the derived linear system. We describe in chapter five the shifted Laplace preconditioner and in chapter six the iterative method IDR(s) and the extensions concerning the initial search space and the approximations to the eigenvalues of the system matrix. In chapter seven we apply some of the methods and techniques of the preceding chapters and perform a set of numerical experiments to the car problem. In the last chapter, we draw conclusions concerning the achievements and make a few suggestions for further research.

Chapter 2

Formulation of the problem

Music is a higher revelation than all wisdom and philosophy.
Ludwig van Beethoven

2.1 The wave equation

Sound and, more generally, acoustic waves generate very small disturbances in the ambient pressure that propagate through a medium such as water and air. In order to study the pressure disturbances in air, the wave equation is applied. For the derivation of this wave equation several conservation laws are used. Conservation of an attribute means that it can neither arise nor vanish, but can only move. This implies that the total rate of inflow in any region equals the rate of increase of that attribute within that region. In this section, we follow [17, Ch. 1 and 3] and [18, Ch. 1,2 and 3] and apply the law of conservation of mass to an infinitesimal volume after which the following (continuity) equation can be derived:

$$\frac{\partial}{\partial t}\rho(\mathbf{x},t) + \nabla \cdot \big(\, \rho(\mathbf{x},t)\, \mathbf{v}(\mathbf{x},t)\big) = 0, \tag{2.1}$$

where $\rho(\mathbf{x},t)$ is the fluid density and $\mathbf{v}(\mathbf{x},t)$ is the flow velocity.

Under the assumption that air is incompressible and has zero viscosity the Euler equation holds. This equation represents conservation of momentum and energy and equates the rate of change of fluid momentum to the negative of the pressure gradient. The Euler equation is given by

$$\nabla p(\mathbf{x},t) + \rho\left(\frac{\partial}{\partial t}\mathbf{v}(\mathbf{x},t) + (\mathbf{v}(\mathbf{x},t)\cdot\nabla)\mathbf{v}(\mathbf{x},t)\right) = \mathbf{0}, \tag{2.2}$$

where $p(\mathbf{x},t)$ is the fluid pressure.

Fluctuations in density and pressure caused by the propagation of even the loudest sounds are more than 1000 times less than the average density ρ_0 (≈ 1.2 kg/m^3) and pressure p_0 ($\approx 10^5$ Pa) of atmospheric air. In addition, these fluctuations occur very quickly since the sound frequencies f that are audible to the average human range from 20 Hz to 20 kHz and therefore we assume that density and pressure fluctuations are adiabatic (which means that there is no heat transfer). Under these conditions we can assume that the change of pressure $\tilde{p}$ due to sound propagation is related only to the change of density $\tilde{\rho}$, that is, $\tilde{p} = \tilde{p}(\tilde{\rho})$. Since the changes are very small compared to the average values of density and pressure, the density and pressure are related linearly and based on the assumption that air is an ideal gas we have that

$$\tilde{p} = c_0^2\tilde{\rho}, \tag{2.3}$$

where c_0 is the speed of sound in air, which is equal to 343 m/s at a temperature of 20°C.

We consider the perturbations $\tilde{\rho}$ around ρ_0 and $\tilde{p}$ around p_0, that is,

$$\rho(\mathbf{x},t) = \rho_0 + \tilde{\rho}(\mathbf{x},t) \text{ and } p(\mathbf{x},t) = p_0 + \tilde{p}(\mathbf{x},t).$$

We substitute these in (2.1) and (2.2) and discard all nonlinear terms (assuming that $\nabla \cdot (\tilde{\rho}\mathbf{v})$ and $(\mathbf{v} \cdot \nabla)\mathbf{v}$ are small), after which we obtain

$$\frac{\partial}{\partial t}\tilde{\rho}(\mathbf{x},t) + \rho_0 \, \nabla \cdot \mathbf{v}(\mathbf{x},t) = 0, \tag{2.4}$$

and

$$\nabla \tilde{p}(\mathbf{x},t) + \rho_0 \, \frac{\partial}{\partial t}\mathbf{v}(\mathbf{x},t) = \mathbf{0}. \tag{2.5}$$

For the remainder of the report we drop the tildes for notational convenience.

We take the difference between the time derivative of (2.4) and the divergence of (2.5) to obtain

$$\left(\frac{\partial^2}{\partial t^2}\rho(\mathbf{x},t) + \rho_0 \nabla \cdot \frac{\partial}{\partial t}\mathbf{v}(\mathbf{x},t)\right) - \left(\nabla^2 p(\mathbf{x},t) + \rho_0 \nabla \cdot \frac{\partial}{\partial t}\mathbf{v}(\mathbf{x},t)\right) = 0,$$

and combine this result with (2.3), which leads to the desired wave equation for to acoustic pressure fluctuation:

$$\nabla^2 p(\mathbf{x},t) = \frac{1}{c_0^2}\frac{\partial^2}{\partial t^2} p(\mathbf{x},t). \tag{2.6}$$

We stress that the sound pressure p in this equation is the acoustic pressure perturbation about the undisturbed pressure p_0.

A solution for the one-dimensional equivalent of (2.6) is

$$p(x,t) = f_r(c_0 t - x) + f_l(c_0 t + x), \tag{2.7}$$

where f_r is a wave traveling to the right and f_l a wave traveling to the left. Waves that are created by sinusoidally vibrating sources in space and time (such as a speaker or a string instrument) with an angular frequency $\omega = 2\pi f$ (where f is the ordinary frequency) result in a pressure perturbation equal to

$$p(x,t) = A_r \sin(\omega t - kx + \varphi_r) + A_l \sin(\omega t + kx + \varphi_l),$$

where A_r and A_l are the amplitudes, φ_r and φ_l are the phase angles and k is the acoustic wavenumber. From (2.7) if follows that $k = \omega/c_0$. Since the wave is sinusoidal, it repeats itself after the wavelength λ and we have $k = 2\pi/\lambda$. We combine these two equation and obtain $\lambda = c_0/f$.

As the wave passes, the air particles oscillate back and forth in the direction of this wave, but there is no net movement of the particles. The disturbances caused by the sound wave travel with a constant speed c_0 and it can be shown [18, Ch. 2.3] that for a wave (traveling to the right) the particle velocity v satisfies

$$\frac{p}{v} = \rho c_0 = Z_0,$$

and Z_0 is the so-called characteristic acoustic impedance ($Z_0 = 415$ kg/(s m^2) in air of 20°C).

2.2 The Helmholtz equation

In the analysis of acoustical problems it can be useful to consider one single frequency. To study a single frequency we assume that $p(\mathbf{x},t) = P(\mathbf{x})\Theta(t)$ and apply separation of variables. Substitution in the wave equation (2.6) results after some rearranging in

$$\frac{1}{P(\mathbf{x})}\nabla^2 P(\mathbf{x}) = \frac{1}{c_0^2\Theta(t)}\frac{d^2}{dt^2}\Theta(t).$$

Since the left hand side of the equation depends on $\mathbf{x}$ only and the right hand side on t only, this equation is valid if and only if both sides are equal to a constant value $-k^2$, which is chosen in this way for convenience. Note that we have defined earlier k as the wavenumber. We obtain the following two equations

$$\nabla^2 P(\mathbf{x}) + k^2 P(\mathbf{x}) = 0, \tag{2.8}$$

which is known as the frequency-domain wave equation or Helmholtz equation, and

$$\frac{d^2}{dt^2}\Theta(t) + \omega^2\Theta(t) = 0, \tag{2.9}$$

where we have used that $\omega = kc_0$.

The fundamental solutions of (2.9) can be written as

$$\Theta_1(t) = ae^{i\omega t} \quad \text{and} \quad \Theta_2(t) = \bar{a}e^{-i\omega t},$$

where $a \in \mathbb{C}$ is a constant that describes the amplitude $|a|$ and the relative phase φ_a of the wave. If $\tilde{P}(\mathbf{x})$ is a solution to (2.8), then $c\tilde{P}(\mathbf{x})$ ($c \in \mathbb{C}$) is also a solution and hence we can absorb the constant a in c and we state that

$$p(\mathbf{x},t) = P(\mathbf{x})\, e^{i\omega t} \tag{2.10}$$

or

$$p(\mathbf{x},t) = P(\mathbf{x})\, e^{-i\omega t},$$

where $P(\mathbf{x})$ satisfies (2.8). We emphasise that the space-dependent solution $P(\mathbf{x})$ holds for the angular frequency $\omega = kc_0$ only.

2.3 Acoustic source term

A differential equation analogous to (2.6) can be obtained when a mass source term is added to the mass conservation equation (2.1) or a force density is added to the Euler equation (2.2). This results by a similar derivation (under certain conditions) to the inhomogeneous wave equation

$$\frac{1}{c_0^2}\frac{\partial^2}{\partial t^2}p(\mathbf{x},t) - \nabla^2 p(\mathbf{x},t) = s(\mathbf{x},t),$$

where $s(\mathbf{x},t)$ is the source term. We consider a harmonic point source which is located at $\mathbf{x}_s$ with angular frequency ω and amplitude $a \neq 0$ so that

$$s(\mathbf{x},t) = ae^{i\omega t}\delta(\mathbf{x}-\mathbf{x}_s).$$

Under these conditions separation of variables can be applied and this results in the Helmholtz equation with source term, which equals

$$-k^2 P(\mathbf{x}) - \nabla^2 P(\mathbf{x}) = S(\mathbf{x}),$$

with $S(\mathbf{x}) = a\delta(\mathbf{x}-\mathbf{x}_s)$. The problem is generally scaled such that $a = 1$.

2.4 Room acoustics and boundary conditions

We consider sound propagation in an enclosed space $\Omega \subset \mathbb{R}^n$, where the sound propagates according to the acoustic wave equation (2.6). Under the assumption that the time dependence of the pressure behaves as $\exp(i\omega t)$, the Helmholtz equation (2.8) holds, which results in a solution of the form (2.10). Both $p(\mathbf{x}, t)$ and $P(\mathbf{x})$ have to meet the characteristics of the boundary Γ that encloses the space.

The characteristics concerning the acoustics of a boundary Γ are expressed in the wall impedance $\widetilde{Z}$, which is the ratio of the acoustic pressure that acts on a point on the boundary and the normal component $v_n = \mathbf{n} \cdot \mathbf{v}$ of the air velocity $\mathbf{v}$ at that point:

$$\widetilde{Z} = \left(\frac{p}{v_n} \right)_\Gamma .$$

Under the assumption that the direction in which the acoustic waves are traveling is normal to the (entire) boundary, we have that $v_n = \|\mathbf{v}\|$ and we obtain the normal acoustic impedance

$$\widetilde{Z}_n = \left(\frac{p}{\|\mathbf{v}\|} \right)_\Gamma .$$

We define the specific normal acoustic impedance Z_n as the ratio of the normal acoustic impedance of the wall and the acoustic impedance of the air, that is,

$$Z_n = \frac{\widetilde{Z}_n}{Z_0}.$$

The boundary conditions for the wave equation (2.6) are expressed in the specific normal acoustic impedance and we have that

$$Z_n \frac{\partial}{\partial n} p(\mathbf{x}, t) + \frac{1}{c_0} \frac{\partial}{\partial t} p(\mathbf{x}, t) = 0 \text{ on } \Gamma. \tag{2.11}$$

This Robin boundary condition is valid for a general absorbing boundary. For certain boundaries this can be simplified to a Dirichlet or Neumann boundary condition. For a so-called open boundary Γ_o that consists of a surface with a very low acoustic impedance, we have that $\widetilde{Z}_n \ll Z_0$ and hence $Z_n \approx 0$. This results in the Dirichlet boundary condition

$$p(\mathbf{x}, t) = 0 \text{ on } \Gamma_\mathrm{o}.$$

When the surface of a boundary Γ_r has much higher impedance than the impedance of air, $\widetilde{Z}_n \gg Z_0$, the second term in (2.11) is negligible. This type of boundary is called reflecting and the Neumann boundary condition holds:

$$\frac{\partial}{\partial n} p(\mathbf{x}, t) = 0 \text{ on } \Gamma_\mathrm{r}.$$

In order to obtain the boundary conditions of the Helmholtz equation (2.8), we substitute (2.10) into (2.11) and we obtain

$$\begin{aligned} &Z_n \frac{\partial}{\partial n} P(\mathbf{x}) + ikP(\mathbf{x}) = 0 && \text{on } \Gamma, \\ &P(\mathbf{x}) = 0 && \text{on } \Gamma_\mathrm{o}, \\ &\frac{\partial}{\partial n} P(\mathbf{x}) = 0 && \text{on } \Gamma_\mathrm{r}. \end{aligned}$$

Chapter 3

Discretisation

Mathematics is the queen of sciences and arithmetic is the queen of mathematics.
Carl Friedrich Gauß

We apply the finite element method and Galerkin method as described in [13, Ch. 5,6 and 7], in order to discretise and numerically solve the Helmholtz equation with a harmonic point source term in some enclosed space Ω, that is, we apply it to

$$-k^2 P(\mathbf{x}) - \nabla^2 P(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_s) \text{ on } \Omega. \tag{3.1}$$

On the boundary $\Gamma = \Gamma_\mathrm{a} \cup \Gamma_\mathrm{o} \cup \Gamma_\mathrm{r}$, the boundary conditions

$$\begin{aligned} &Z_n \frac{\partial}{\partial n} P(\mathbf{x}) + ikP(\mathbf{x}) = 0 \quad &&\text{on } \Gamma_\mathrm{a}, \\ &P(\mathbf{x}) = 0 &&\text{on } \Gamma_\mathrm{o}, \\ &\frac{\partial}{\partial n} P(\mathbf{x}) = 0 &&\text{on } \Gamma_\mathrm{r}, \end{aligned} \tag{3.2}$$

are imposed.

We note that the solution $P(\mathbf{x})$ of the weak form must be in the space $\Sigma = \{P \in H^1(\Omega) \ : \ P|_{\Gamma_\mathrm{o}} = 0\}$, where $H^1(\Omega)$ is the first Sobolev space.

We multiply the Helmholtz equation (3.1) with an arbitrary testfunction $\eta \in \Sigma$ and integrate over Ω to obtain

$$-k^2 \int_\Omega \eta P \, d\Omega - \int_\Omega \eta \nabla^2 P \, d\Omega = \int_\Omega \eta \delta(\mathbf{x} - \mathbf{x}_s) \, d\Omega.$$

We apply Greens first identity,

$$\int_\Omega \psi \nabla^2 \varphi \, d\Omega + \int_\Omega \nabla\varphi \cdot \nabla\psi \, d\Omega = \oint_\Gamma \psi \frac{\partial \varphi}{\partial n} \, d\Gamma,$$

to the second term of the left hand side and obtain

$$-k^2 \int_\Omega \eta P \, d\Omega + \int_\Omega \nabla P \cdot \nabla \eta \, d\Omega - \oint_\Gamma \eta \frac{\partial P}{\partial n} \, d\Gamma = \int_\Omega \eta \delta(\mathbf{x} - \mathbf{x}_s) \, d\Omega.$$

Substitution of the boundary conditions (3.2) results in

$$-k^2 \int_\Omega \eta P \, d\Omega + \int_\Omega \nabla P \cdot \nabla \eta \, d\Omega + ik \int_{\Gamma_\mathrm{a}} \frac{1}{Z_n} \eta P \, d\Gamma = \int_\Omega \eta \delta(\mathbf{x} - \mathbf{x}_s) \, d\Omega. \tag{3.3}$$

We solve the equation (3.1) with boundary conditions (3.2) by finding a solution of the equivalent weak form, that is, by finding $P \in \Sigma$ such that (3.3) is satisfied for all $\eta \in \Sigma$. We note that such a

P automatically satisfies the natural boundary condition $\partial P/\partial n = 0$ on Γ_r. The essential boundary condition $P = 0$ on Γ_o is met too since we defined earlier that $\Sigma = \{P \in H^1(\Omega) \ : \ P|_{\Gamma_\mathrm{o}} = 0\}$.

We use the finite element method and subdivide Ω into N simplex-shaped elements that consists of n distinct nodes (corner points) $\mathbf{x}^j$ $(j = 1, \ldots, n)$. We choose n basis functions $\eta_i(\mathbf{x}) \in \Sigma$ $(i = 1, \ldots, n)$ such that $\eta_i(\mathbf{x})$ is linear per simplex and the i-th basis function is equal to 1 at node $\mathbf{x}^i$ and 0 at all others, that is, $\eta_i(\mathbf{x}^j) = \delta_{ij}$. We refer to Figure 3.1 for an example of such a piecewise linear basis function on (a part of) a two-dimensional triangular mesh.

It is needed that the position of the source coincides with a node since if and only if the source $\mathbf{x}_s$ corresponds to the k-th node we have that

$$\int_\Omega \delta(\mathbf{x} - \mathbf{x}_s)\eta_i(\mathbf{x}) \ d\Omega = \begin{cases} 1 & i = k, \\ 0 & i \neq k. \end{cases}$$

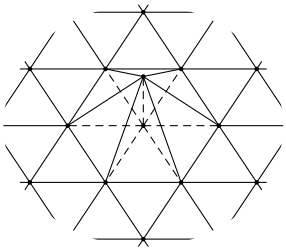


Figure 3.1: A piecewise linear basis function.

We apply the method of Galerkin by approximating the solution $P(\mathbf{x})$ by a finite linear combination of the n piecewise linear basis functions and state that there exist coefficients p_i $(i = 1, \ldots, n)$ such that

$$P(\mathbf{x}) \approx P^n(\mathbf{x}) = \sum_{i=1}^{n} p_i \eta_i(\mathbf{x}).$$

We choose $\eta = \eta_j$ $(j = 1, \ldots, n)$ and we obtain n linear equations, since for each $j = 1, \ldots, n$ we have that

$$\sum_{\ell=1}^{n} p_\ell \left(\int_\Omega \nabla\eta_\ell \cdot \nabla\eta_j \ d\Omega + ik \int_{\Gamma_\mathrm{a}} \frac{1}{Z_0} \eta_\ell \eta_j \ d\Gamma - k^2 \int_\Omega \eta_\ell \eta_j \ d\Omega \right) = \int_\Omega \eta_j \delta(\mathbf{x} - \mathbf{x}_s) \ d\Omega.$$

This last equation can be written as

$$(\mathbf{K} + ik\mathbf{C} - k^2\mathbf{M})\mathbf{p} = \mathbf{b}, \tag{3.4}$$

and a $P(\mathbf{x})$ that satisfies the weak formulation (3.3) can be approximated by the solution of the linear system of equations (3.4). The right hand side of this equation is equal to the k-th standardunit vector $\mathbf{b} = \mathbf{e}_k$.

The stiffness matrix $\mathbf{K}$, the matrix $\mathbf{C}$ and the mass matrix $\mathbf{M}$ are all real, symmetric and positive semi definite. The system matrix $(\mathbf{K} + ik\mathbf{C} - k^2\mathbf{M})$ is complex, symmetric and indefinite. (A real matrix $\mathbf{A}$ is positive semi definite or psd if $\mathbf{x}^T\mathbf{A}\mathbf{x} \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$ and indefinite if both $\mathbf{A}$ and $-\mathbf{A}$ are not psd.)

3.1 Example

As an example we consider a two-dimensional room of L by L meters $\Omega = [0, L] \times [0, L]$ surrounded by four walls $\Gamma = \Gamma_n \cup \Gamma_e \cup \Gamma_s \cup \Gamma_w$. The wall at the east Γ_e consists of a sound absorbing material with $Z_n = \frac{1}{5} - \frac{3}{2}i$ kg / (s m^2) while the other three walls $\Gamma_n, \Gamma_s, \Gamma_w$ reflect the sound and hence satisfy a Neumann boundary condition. We fix $L = 4$ m and locate an harmonic point source at the centre of the room: $\mathbf{x}_s = (2, 2)^T$.

We determine the propagation of sound with frequency $f = kc_0/2\pi$ in this room by solving the following Helmholtz equation with boundary conditions:

$$\begin{aligned} &\nabla^2 P(\mathbf{x}) + k^2 P(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_s) && \text{on } \Omega, \\ &Z_n \frac{\partial}{\partial n} P(\mathbf{x}) + ikP(\mathbf{x}) = 0 && \text{on } \Gamma_e, \\ &\frac{\partial}{\partial n} P(\mathbf{x}) = 0 && \text{on } \Gamma_n, \Gamma_s, \Gamma_w. \end{aligned} \tag{3.5}$$

The weak form of this problem is equal to

$$\int_\Omega \nabla P \cdot \nabla \eta \; d\Omega + ik \int_{\Gamma_e} \frac{1}{Z_n} \eta \; d\Gamma - k^2 \int_\Omega P\eta \; d\Omega = \int_\Omega \delta(\mathbf{x} - \mathbf{x}_s)\eta \; d\Omega, \tag{3.6}$$

where $\eta \in H^1(\Omega)$ is an arbitrary test function.

The room Ω is divided into triangular elements. See Figure 3.2 for an example of a coarse subdivision. In this particular subdivision we have chosen a gridsize $h = L/12$ m, which means that we have $n = (L/h+1)^2 = 169$ nodes $\mathbf{x}^i$ and $N = 2 \cdot (L/h)^2 = 288$ elements. The absorbing boundary Γ_e includes the nodes $\mathbf{x}^{157}, \ldots, \mathbf{x}^{169}$ and $\mathbf{x}_s = \mathbf{x}^{\lceil n/2 \rceil}$ since the source is at the centre of the room.

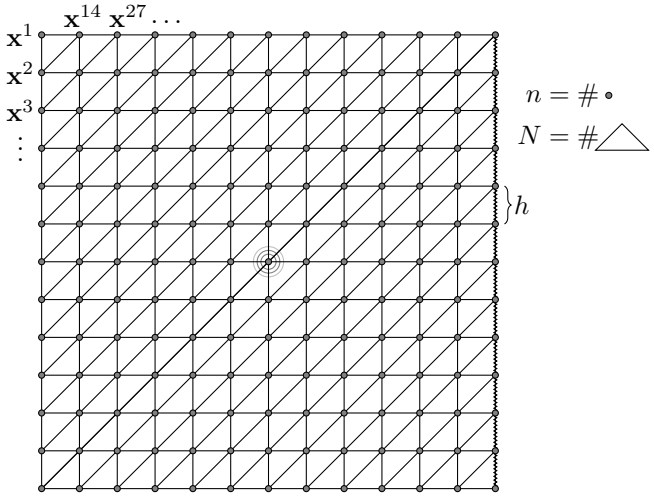


Figure 3.2: A subdivision of Ω into triangular elements.

The Galerkin method applied to (3.6) with the basis functions η_i results in n linear equations

$$\sum_{\ell=1}^{n} p_\ell \left(\int_\Omega \nabla \eta_\ell \cdot \nabla \eta_j d\Omega + ik \int_{\Gamma_e} \frac{1}{Z_n} \eta_j \; d\Omega - k^2 \int_\Omega \eta_\ell \eta_j \; d\Omega \right) = \int_\Omega \delta(\mathbf{x} - \mathbf{x}_s)\eta \; d\Omega \; (j = 1, \ldots, n),$$

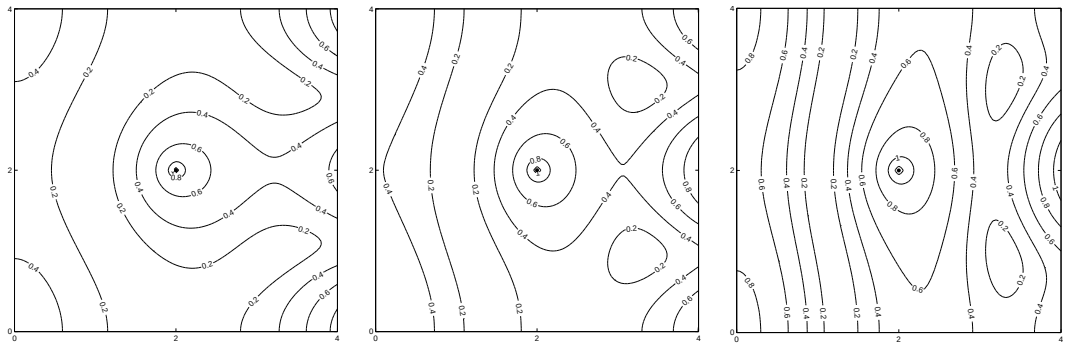
which can equivalently be written as the linear system

$$(\mathbf{K} + ik\mathbf{C} - k^2\mathbf{M})\mathbf{p} = \mathbf{b}. \tag{3.7}$$

The matrices $\mathbf{K}$, $\mathbf{C}$ and $\mathbf{M}$ are very sparse. If the Newton-Cotes quadrature rules are used for the numerical integration, we obtain the respective structures

$$\mathbf{K} = \begin{pmatrix} \ddots & \ddots & \ddots & & \\ \ddots & \ddots & \ddots & \ddots & \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ & \ddots & \ddots & \ddots & \ddots \\ & & \ddots & \ddots & \ddots \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \ddots \end{pmatrix}, \quad \mathbf{M} = \begin{pmatrix} \ddots & & & & \\ & \ddots & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & \ddots \end{pmatrix}.$$

The solution to (3.7), which is a numerical approximation of the solution to (3.5), for $f = 70$ Hz, $f = 72$ Hz and $f = 74$ Hz is given in Figure 3.3. It is clear that the solutions differ significantly even for frequencies that are close to each other. We can also see from this figure that $\frac{\partial}{\partial n}|P| = 0$ for all walls, except for the wall at the east.



(a) $f = 70$ Hz (b) $f = 72$ Hz (c) $f = 74$ Hz

Figure 3.3: Contour lines of the amplitude $|P(\mathbf{x})|$.

Chapter 4

Iterative methods

Any sufficiently advanced technology is indistinguishable from magic.
Sir Arthur Charles Clarke

Modelling the propagation of acoustic waves in some space Ω results in a large, sparse and linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{4.1}$$

with $\mathbf{A} = (a_{ij})^{n\times n} \in \mathbb{C}^{n\times n}$ a square coefficient matrix and $\mathbf{b} \in \mathbb{C}^n$ the right hand side vector. The objective is to determine the solution vector $\mathbf{x} \in \mathbb{C}^n$ that satisfies (4.1). Various numerical methods have been developed for solving these type of linear systems. An overview of iterative methods can be found in [16] and [21] and these books have served as the basis for the upcoming sections.

Standard or direct methods compute the solution in a finite number of steps by using a matrix factorisation or decomposition and give the exact answer if they are performed with infinite precision arithmetic. Since finite precision is used the result is an approximation of the exact solution. An example of a direct method is the LU factorisation, which factors the matrix $\mathbf{A}$ as the product of a lower triangular matrix $\mathbf{L}$ and an upper triangular matrix $\mathbf{U}$ such that $\mathbf{LU} = \mathbf{A}$, after which the problem (4.1) can be solved directly by forward and backward substitution.

Since direct methods scale poorly with the size of the problem, which means that it is very time consuming to determine for instance the LU factors for large $\mathbf{A}$, iterative methods are developed. These methods generate a sequence of improving approximations $\mathbf{x}_0, \mathbf{x}_1, \ldots$ of the desired solution $\mathbf{x}$ until a termination criterium is satisfied. The sequence $\{\mathbf{x}_i\}_{i\geq 0}$ converges to $\mathbf{x}$ in some norm $\|\cdot\|$ (which is usually the 2-norm) for a given initial guess $\mathbf{x}_0$ if and only if

$$\lim_{i\to\infty} \|\mathbf{x} - \mathbf{x}_i\| = 0.$$

The vector $\boldsymbol{\epsilon}_i = \mathbf{x} - \mathbf{x}_i$ is known as the i-th error. Since $\mathbf{x}$ is unknown, the error $\boldsymbol{\epsilon}_i$ is not available and hence the residuals $\mathbf{r}_i = \mathbf{b} - \mathbf{A}\mathbf{x}_i$ are used. Note that the residuals and the errors are closely related since $\mathbf{r}_i = \mathbf{A}(\mathbf{x} - \mathbf{x}_i) = \mathbf{A}\boldsymbol{\epsilon}_i$.

4.1 Krylov space methods

The most simple Krylov subspace method is the standard Richardson iteration. In this case, we have the recursion

$$\mathbf{x}_i = \mathbf{x}_{i-1} + (\mathbf{b} - \mathbf{A}\mathbf{x}_{i-1}) = \mathbf{x}_{i-1} + \mathbf{r}_{i-1}.$$

Multiplying the above equation with $-\mathbf{A}$ results in $\mathbf{r}_i = (\mathbf{I} - \mathbf{A})\mathbf{r}_{i-1}$, so

$$\mathbf{r}_i = (\mathbf{I} - \mathbf{A})^i \mathbf{r}_0.$$

It follows from these equations that

$$\begin{aligned}
\mathbf{x}_1 &= \mathbf{x}_0 + \mathbf{r}_0, \\
\mathbf{x}_2 &= \mathbf{x}_1 + \mathbf{r}_1 = \mathbf{x}_0 + \mathbf{r}_0 + (\mathbf{I} - \mathbf{A})\mathbf{r}_0, \\
\mathbf{x}_3 &= \mathbf{x}_2 + \mathbf{r}_2 = \mathbf{x}_0 + \mathbf{r}_0 + (\mathbf{I} - \mathbf{A})\mathbf{r}_0 + (\mathbf{I} - \mathbf{A})^2\mathbf{r}_0, \\
&\vdots
\end{aligned}$$

which means that the approximations $\mathbf{x}_i$ satisfy

$$\mathbf{x}_i = \mathbf{x}_0 + \sum_{j=0}^{i} (\mathbf{I} - \mathbf{A})^j \mathbf{r}_0. \tag{4.2}$$

This implies that if $\rho(\mathbf{I} - \mathbf{A}) < 1$ we have that $\mathbf{A}$ is non-singular and

$$\begin{aligned}
\lim_{i \to \infty} \mathbf{x}_i &= \mathbf{x}_0 + \sum_{j=0}^{\infty} (\mathbf{I} - \mathbf{A})^j \mathbf{r}_0 \\
&= \mathbf{x}_0 + [\mathbf{I} - (\mathbf{I} - \mathbf{A})]^{-1} \mathbf{r}_0 \\
&= \mathbf{x}_0 + \mathbf{A}^{-1} \mathbf{r}_0 \\
&= \mathbf{A}^{-1}\mathbf{b} \;=\; \mathbf{x}.
\end{aligned}$$

It also follows that $\mathbf{x}_i$ lies in the affine space $\mathbf{x}_0 + \text{span}\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \ldots, \mathbf{A}^{i-1}\mathbf{r}_0\}$. The subspace

$$\mathcal{K}_i(\mathbf{A}, \mathbf{r}_0) = \text{span}\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \ldots, \mathbf{A}^{i-1}\mathbf{r}_0\}$$

is known as the Krylov subspace of dimension i, generated by matrix $\mathbf{A}$ from initial residual $\mathbf{r}_0$.

Since the geometric series (4.2) converges very slowly it is reasonable to consider some other polynomial than $P_i(\xi) = \sum_{j=0}^{i} (1 - \xi)^j$ in $\mathbf{A}$. Methods that use this general polynomial approach are known as Krylov space methods: a standard Krylov space method solves (4.1) with some initial guess $\mathbf{x}_0$ iteratively by generating a sequence $\{\mathbf{x}_i\}$ such that

$$\mathbf{x}_i - \mathbf{x}_0 = P_{i-1}(\mathbf{A})\mathbf{r}_0 \in \mathcal{K}_i(\mathbf{A}, \mathbf{r}_0),$$

with $P_{i-1}(\xi)$ some polynomial (with degree $i-1$). The residuals $\mathbf{r}_i$ satisfy the polynomial relation $\mathbf{r}_i = R_i(\mathbf{A})\mathbf{r}_0$, with

$$R_i(\xi) = 1 - \xi P_{i-1}(\xi).$$

We will describe some Krylov subspace methods in the next sections.

4.2 Arnoldi iteration

The Arnoldi iteration was proposed in 1951 by W.E. Arnoldi and published in [1] and is used to build an orthonormal basis of the k-th Krylov space $\mathcal{K}_k(\mathbf{A}, \mathbf{r}_0)$, which means that it determines orthonormal vectors $\mathbf{q}_1, \ldots, \mathbf{q}_k$ such that

$$\text{span}\{\mathbf{q}_1, \ldots, \mathbf{q}_k\} = \mathcal{K}_k(\mathbf{A}, \mathbf{r}_0).$$

The matrix $\mathbf{Q}_k = (\mathbf{q}_1, \ldots, \mathbf{q}_k)$ satisfies $\mathbf{Q}_k \mathbf{Q}_k^T = \mathbf{I}$. We create $\mathbf{Q}_k$ by starting with the initial unit vector $\mathbf{q}_1 = \mathbf{r}_0/\|\mathbf{r}_0\|$, after which we recursively compute vectors the $\mathbf{q}_j$ $(j = 2, 3, \ldots, k)$ by

$$\mathbf{q}_j = \frac{\mathbf{A}\mathbf{q}_{j-1} - \sum_{i=1}^{j-1} h_{i,j-1}\mathbf{q}_i}{h_{j,j-1}}, \tag{4.3}$$

where $h_{i,j} = \mathbf{A}\mathbf{q}_i \cdot \mathbf{q}_j$ and $h_{j,j-1} = \|\mathbf{A}\mathbf{q}_{j-1} - \sum_{i=1}^{j-1} h_{i,j-1}\mathbf{q}_i\|$. So the Arnoldi iteration multiplies the earlier obtained vector $\mathbf{q}_j$ by $\mathbf{A}$ and orthonormalises the result against all previously obtained $\mathbf{q}_i$, $i = 1, \ldots, j-1$. The method of orthonormalisation is known as the Gram-Schmidt process.

We rewrite (4.3) as $h_{1,j-1}\mathbf{q}_1 + \cdots + h_{j,j-1}\mathbf{q}_j = \mathbf{A}\mathbf{q}_{j-1}$ and after combining these equations for all $\mathbf{q}_i$ with $i < j$, we obtain the matrix equation

$$\mathbf{Q}_j \underline{\mathbf{H}}_{j-1} = \mathbf{A}\mathbf{Q}_{j-1}, \tag{4.4}$$

where the entries of $\underline{\mathbf{H}}_{j-1}^T \in \mathbb{C}^{j\times(j-1)}$ are uniquely determined by the coefficients $h_{i,j}$ (and hence has zeros at $h_{i,j}$ for all $i > j+1$). After k iterations we obtain $h_{k+1,k} = 0$ and hence $\mathbf{q}_{k+1}$ cannot be determined and we obtain after deleting the bottom row of (4.4)

$$\mathbf{Q}_k \mathbf{H}_k = \mathbf{A}\mathbf{Q}_k,$$

or equivalently $\mathbf{A} = \mathbf{Q}_k \mathbf{H}_k \mathbf{Q}_k^T$, with $\mathbf{H}_k \in \mathbb{C}^{k\times k}$ an upper Hessenberg matrix, a matrix that has zero entries below the first subdiagonal.

Note that the algorithm does not break down as long as the dimension of $\mathcal{K}_k(\mathbf{A}, \mathbf{r}_0)$ equals k. The set of vectors $\{\mathbf{q}_1, \ldots, \mathbf{q}_k\}$ indeed form an orthonormal basis of the Krylov subspace $\mathcal{K}_k(\mathbf{A}, \mathbf{r}_0)$ since $\mathbf{q}_1 = \mathbf{r}_0/\|\mathbf{r}_0\|$ and each vector $\mathbf{q}_j$ can be written as $R_j(\mathbf{A})\mathbf{q}_1$ (with $R_j(\xi)$ some polynomial of degree j). An algorithmic description of the Arnoldi iteration is given below.

Algorithm 1 Arnoldi iteration.

Require: $\mathbf{q}_1$ such that $\|\mathbf{q}_1\| = 1$
  for $j = 1, \ldots, k$ **do**
    $\mathbf{v} = \mathbf{A}\mathbf{q}_j$
    for $i = 1, \ldots, j$ **do**
      $h_{i,j} = \mathbf{v} \cdot \mathbf{q}_i$
      $\mathbf{v} = \mathbf{v} - h_{i,j}\mathbf{q}_i$
    end for
    $h_{j+1,j} = \|\mathbf{v}\|$
    $\mathbf{q}_{j+1} = \mathbf{v}/h_{j+1,j}$
  end for

4.3 Generalised Minimal Residual

The Generalised Minimal Residual method (GMRES) was proposed by Y. Saad and M.H. Schultz in 1986 in [19] and is applicable to systems (4.1) when $\mathbf{A}$ is a general (non-singular) square matrix. GMRES makes use of the Arnoldi iteration and minimises in the k-th iteration the residual norm $\|\mathbf{r}_k\|$ over all vectors in $\mathbf{x}_0 + \mathcal{K}_k(\mathbf{A}, \mathbf{r_0})$ by solving the least square problem

$$\min_{\mathbf{x}_i \in \mathbf{x}_0 + \mathcal{K}_k} \|\mathbf{r}_k\| = \min_{\mathbf{x}_i \in \mathbf{x}_0 + \mathcal{K}_k} \|\mathbf{b} - \mathbf{A}\mathbf{x}_k\|. \tag{4.5}$$

In fact, GMRES solves an equivalent but smaller least square problem that results in a vector $\mathbf{y}_k \in \mathbb{C}^k$ that satisfies

$$\mathbf{x}_k = \mathbf{x}_0 + \mathbf{Q}_k \mathbf{y}_k, \tag{4.6}$$

and $\mathbf{x}_k \in \mathbf{x}_0 + \mathcal{K}_k(\mathbf{A}, \mathbf{r_0})$ since $\mathbf{q}_1, \ldots, \mathbf{q}_k$ is an orthonormal basis for $\mathcal{K}_k(\mathbf{A}, \mathbf{r_0})$.

We derive this smaller least square problem by substituting (4.6) and the Arnoldi relation (4.4) into (4.5) and

$$\begin{aligned}
\mathbf{r}_k &= \mathbf{b} - \mathbf{A}\mathbf{x}_k \\
&= \mathbf{b} - \mathbf{A}(\mathbf{x}_0 + \mathbf{Q}_k \mathbf{y}_k) \\
&= \mathbf{r}_0 - \mathbf{A}\mathbf{Q}_k \mathbf{y}_k \\
&= \|\mathbf{r}_0\|\mathbf{q}_1 - \mathbf{Q}_{k+1}\underline{\mathbf{H}}_k \mathbf{y}_k \\
&= \mathbf{Q}_{k+1}(\|\mathbf{r}_0\|\mathbf{e}_1 - \underline{\mathbf{H}}_k \mathbf{y}_k).
\end{aligned}$$

Since $\mathbf{Q}_{k+1}$ is a matrix with orthonormal columns, we have that

$$\|\mathbf{b} - \mathbf{A}\mathbf{x}_k\| = \|\beta \mathbf{e}_1 - \underline{\mathbf{H}}_k \mathbf{y}_k\|,$$

where $\beta = \|\mathbf{r}_0\|$. So the solution to the minimisation problem (4.5) can be obtained by solving the smaller least square problem

$$\min_{\mathbf{y}_k \in \mathbb{C}^k} \|\beta \mathbf{e}_1 - \underline{\mathbf{H}}_k \mathbf{y}_k\|, \tag{4.7}$$

after which the k-th approximation $\mathbf{x}_k = \mathbf{x}_0 + \mathbf{Q}_k \mathbf{y}_k$ is determined. The above leads to the GMRES algorithm as given in Algorithm 2.

To solve the least square problem (4.7) the QR decomposition is used. It decomposes the Hessenberg matrix $\underline{\mathbf{H}}_k$ as $\widetilde{\mathbf{Q}}_{k+1}^* \underline{\mathbf{H}}_k = \underline{\mathbf{R}}_k$, with $\widetilde{\mathbf{Q}}_{k+1}$ a $(k+1)\times(k+1)$ orthogonal matrix, $\underline{\mathbf{R}}_k = (\mathbf{R}_k^T, \mathbf{0})^T$, and $\mathbf{R}_k$ an upper triangular matrix. This is done by left multiplication of (4.7) with a sequence of Givens matrices $\mathbf{\Phi}_i$ of size $(k+1)\times(k+1)$, (with $i = 1, \ldots, k$). The non-zero elements of these matrices are

$$\begin{aligned}
\phi_{j,j} &= 1 \ (j \neq i, i+1), \\
\phi_{i,i} &= \phi_{i+1,i+1} = \cos(\theta_i) = h_{i,i}/(h_{i,i}^2 + h_{i+1,i}^2)^{1/2}, \\
\phi_{i,i+1} &= -\phi_{i+1,i} = \sin(\theta_i) = h_{i+1,i}/(h_{i,i}^2 + h_{i+1,i}^2)^{1/2}.
\end{aligned}$$

Note that the coefficient θ_i of $\mathbf{\Phi}_i$ is chosen in such a way that $h_{i+1,i}$ is eliminated after multiplication with $\mathbf{\Phi}_i$. Since all $\mathbf{\Phi}_i$ are orthogonal matrices we have that the product $\widetilde{\mathbf{Q}}_{k+1}^* = \mathbf{\Phi}_1 \cdot \mathbf{\Phi}_2 \cdots \mathbf{\Phi}_k$ is orthogonal also and it follows that

$$\|\beta \mathbf{e}_1 - \underline{\mathbf{H}}_k \mathbf{y}_k\| = \|\widetilde{\mathbf{Q}}_{k+1}^*(\beta \mathbf{e}_1 - \underline{\mathbf{H}}_k \mathbf{y}_k)\| = \|\beta \widetilde{\mathbf{Q}}_{k+1}^* \mathbf{e}_1 - \underline{\mathbf{R}}_k \mathbf{y}_k\|.$$

The solution to the minimisation problem (4.7) is obtained by solving the triangular system that remains after deleting the bottom rows of the matrices in the above minimisation problem.

In appears that for some $k \ll n$ the norm of the k-th residual is almost always already very small, which means that the approximation $\mathbf{x}_k$ of the solution $\mathbf{x}$ is sufficiently accurate and only k iterations are needed. In addition, the minimisation problem (4.7) is much easier to solve than the equivalent problem (4.5) since $\underline{\mathbf{H}}_k$ is $(k+1)\times k$ while $\mathbf{A}$ is $n \times n$. Also, it is shown in [19] that $h_{k+1,k}|\mathbf{e}_k \cdot \mathbf{y}_k| = \|\mathbf{b} - \mathbf{A}\mathbf{x}_k\| = \|\mathbf{r}_k\|$, so the norm of the k-th residual can be obtained easily.

These characteristics ensure that GMRES is applicable to certain large problems.

A disadvantage of GMRES is that the amount of work and memory that is needed to store and compute $\underline{\mathbf{H}}_k$ and $\mathbf{Q}_k$ increases with j. This long recurrence is sufficient to make GMRES unworkable for problems that need a lot of iterations. GMRES(m) is developed to provide a solution for the memory and work problems. The algorithm performs the same steps, but restarts after m iterations once $\mathbf{x}_0$ is set equal to $\mathbf{x}_m$. However, this can cause stagnation if $\mathbf{A}$ is not psd.

Algorithm 2 Generalised Minimal Residual.

Require: $\mathbf{x}_0$, tol $\in (0,1)$, maxit > 0
  $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\beta_0 = \|\mathbf{r}_0\|$ and $\mathbf{q}_1 = \mathbf{r}_0/\beta_0$, $j = 0$
  while $\beta_j >$ tol and $j \leq$ maxit **do**
    $j = j + 1$
    $\mathbf{v} = \mathbf{A}\mathbf{q}_j$
    for $i = 1, \ldots, j$ **do**
      $h_{i,j} = \mathbf{v} \cdot \mathbf{q}_i$
      $\mathbf{v} = \mathbf{v} - h_{i,j}\mathbf{q}_i$
    end for
    $h_{j+1,j} = \|\mathbf{v}\|$
    $\mathbf{q}_{j+1} = \mathbf{v}/h_{j+1,j}$
    solve $\min_{\mathbf{y}_j} \|\beta_0 \mathbf{e}_1 - \underline{\mathbf{H}}_j \mathbf{y}_j\|$
    $\beta_j = h_{j+1,j}|\mathbf{e}_j \cdot \mathbf{y}_j|$
  end while
  $\mathbf{x}_j = \mathbf{x}_0 + \mathbf{Q}_j \mathbf{y}_j$

4.3.1 Convergence

Since GMRES minimises at each iteration the residual $\|\mathbf{r}_k\|$ the method is optimal in finding the solution of (4.1) in terms of number of iterations. This does, however, not at all imply that GMRES is optimal in computation time and storage requirements.

We make two observations. The first is that in infinite precision arithmetic GMRES will always converge in (at most) n steps, which means that $\mathbf{r}_n = \mathbf{0}$ and $\mathbf{x}_n = \mathbf{x}$. This is the case since $\mathcal{K}_n(\mathbf{A}, \mathbf{r}_0) = \mathbb{C}^n$. The second is that $\|\mathbf{r}_{i+1}\| \leq \|\mathbf{r}_i\|$, which means that the convergence is monotonic. This follows from the fact that $\mathcal{K}_i(\mathbf{A}, \mathbf{r}_0) \subset \mathcal{K}_{i+1}(\mathbf{A}, \mathbf{r}_0)$. The first observation is not significant in practice since GMRES is used as an iterative method: we want to know if GMRES converges (to a specified tolerance) in $k \ll n$ iterations.

We know that $\|\mathbf{r}_k\| = \|R_k(\mathbf{A})\mathbf{r}_0\| \leq \|R_k(\mathbf{A})\|\|\mathbf{r}_0\|$ is minimal. Now, the critical factor for the size of this quantity is almost always $\|R_k(\mathbf{A})\|$ and this means that the convergence rate of GMRES is usually determined by the inequality

$$\frac{\|\mathbf{r}_k\|}{\|\mathbf{r}_0\|} \leq \inf_{R_k \in \mathbb{P}_k} \|R_k(\mathbf{A})\|. \tag{4.8}$$

The value of $\|R_k(\mathbf{A})\|$ can be estimated by considering polynomials that satisfy $R_k(0) = 1$ and that are as small as possible on the set of eigenvalues $\sigma(\mathbf{A})$. It follows that convergence depends to a large extent on $\sigma(\mathbf{A})$ and that for fast convergence the eigenvalues need to be clustered away from the origin [21, Ch 6.11.4].

4.4 BiCG, CGS and BiCGSTAB

4.4.1 Bi-Conjugate Gradient

Where GMRES preserves orthogonality of the residuals, the Bi-Conjugate Gradient method (BiCG) considers two sequences of residuals $\{\mathbf{r}_i\}_{i\geq 0}$ and $\{\tilde{\mathbf{r}}_i\}_{i\geq 0}$ that are mutually orthogonal. The method was introduced in 1976 by R. Fletcher in [5]. For the update of the residuals $\tilde{\mathbf{r}}_i$, the conjugate transpose of $\mathbf{A}$ is used and the two updates are

$$\mathbf{r}_i = \mathbf{r}_{i-1} - \alpha_i \mathbf{A}\mathbf{p}_i \quad \text{and} \quad \tilde{\mathbf{r}}_i = \tilde{\mathbf{r}}_{i-1} - \alpha_i \mathbf{A}^* \tilde{\mathbf{p}}_i,$$

where

$$\mathbf{p}_i = \mathbf{r}_{i-1} - \beta_{i-1}\mathbf{p}_{i-1} \quad \text{and} \quad \tilde{\mathbf{p}}_i = \tilde{\mathbf{r}}_{i-1} - \beta_{i-1}\tilde{\mathbf{p}}_{i-1}.$$

In order to meet the orthogonality relations

$$\mathbf{r}_i \cdot \tilde{\mathbf{r}}_j = 0 = \mathbf{p}_i \cdot \mathbf{A}\tilde{\mathbf{p}}_j \quad (i \neq j),$$

the choices for α_i and β_i are

$$\alpha_i = \frac{\tilde{\mathbf{r}}_{i-1} \cdot \mathbf{r}_{i-1}}{\tilde{\mathbf{p}}_i \cdot \mathbf{A}\mathbf{p}_i} \quad \text{and} \quad \beta_i = \frac{\tilde{\mathbf{r}}_i \cdot \mathbf{r}_i}{\tilde{\mathbf{r}}_{i-1} \cdot \mathbf{r}_{i-1}}.$$

Note that BiCG needs two matrix-vector multiplications (MATVECS) per iteration: one with $\mathbf{A}$ and one with $\mathbf{A}^*$, where the conjugate transpose of $\mathbf{A}$ is used only for the computation of α_i and β_i. The methods CGS and BiCGSTAB, which are discussed in the upcoming sections, are derived from BiCG and do not use the matrix multiplication with $\mathbf{A}^*$.

4.4.2 Conjugate Gradient Squared

As with all Krylov subspace methods, the residuals of BiCG satisfy a polynomial relation: $\mathbf{r}_i = R_i(\mathbf{A})\mathbf{r}_0$ and $\tilde{\mathbf{r}}_i = R_i(\mathbf{A})\tilde{\mathbf{r}}_0$. For the determination of the coefficients α_i and β_i we need

$$\rho_i = \tilde{\mathbf{r}}_i \cdot \mathbf{r}_i = R_i(\mathbf{A}^*)\tilde{\mathbf{r}}_0 \cdot R_i(\mathbf{A})\mathbf{r}_0,$$

and this can be written equivalently as

$$\rho_i = \tilde{\mathbf{r}}_0 \cdot R_i^2(\mathbf{A})\mathbf{r}_0,$$

and $\mathbf{A}^*$ is not needed in the determination of α_i and β_i. This leads to the Conjugate Gradient Squared method (CGS), which is derived by P. Sonneveld and published in [26] in 1989. The recursion for the residuals $\mathbf{r}_i^{\mathrm{CGS}}$ is given by

$$\mathbf{r}_i^{\mathrm{CGS}} = R_i^2(\mathbf{A})\mathbf{r}_0.$$

CGS method does not involve computations with $\tilde{\mathbf{r}}_i$ and $\mathbf{A}^*$, but we still need two MATVECS per iteration. The speed of convergence for CGS is in many cases twice as fast as for BiCG, but the convergence behaviour is in general very irregular.

4.4.3 Bi-Conjugate Gradient Stabilised

The Bi-Conjugate Gradient Stabilised method (BiCGSTAB) was presented in [30] in 1992 by H. A. van der Vorst. Instead of computing the residuals with the polynomials $R_i(\mathbf{A})$ (as in BiCG) or $R_i^2(\mathbf{A})$ (as in CGS), BiCGSTAB determines the i-th residual through

$$\mathbf{r}_i^{\text{BiCGSTAB}} = Q_i(\mathbf{A})R_i(\mathbf{A})\mathbf{r}_0,$$

where $Q_i(\mathbf{A}) = (\mathbf{I} - \omega_1\mathbf{A}) \cdot (\mathbf{I} - \omega_2\mathbf{A}) \cdots (\mathbf{I} - \omega_i\mathbf{A})$. The coefficients ω_i are generally chosen in such a way, that the norm of the i-th residual is minimised with respect to ω_j. BiCGSTAB needs also 2 MATVECS per iteration and the convergence is often much faster than BiCG and much smoother than CGS.

4.5 Preconditioning

In order to speed up the convergence and improve the stability of a method, preconditioning is applied. Preconditioning transforms the original linear system into another system with the same solution that has more favorable spectral properties and hence is easier to solve for an iterative method. In applying preconditioning, we replace the original system (4.1) by

$$\mathbf{P}^{-1}\mathbf{A}\mathbf{x} = \mathbf{P}^{-1}\mathbf{b}, \tag{4.9}$$

with $\mathbf{P}$ the (left) preconditioner. The preconditioner is often decomposed as $\mathbf{P} = \mathbf{P}_{\text{L}}\mathbf{P}_{\text{R}}$ and the preconditioned system with the same solution as (4.1) is

$$(\mathbf{P}_{\text{L}}^{-1}\mathbf{A}\mathbf{P}_{\text{R}})(\mathbf{P}_{\text{R}}^{-1}\mathbf{x}) = \mathbf{P}_{\text{L}}^{-1}\mathbf{b}. \tag{4.10}$$

A good preconditioner $\mathbf{P}$ should in general be easy to construct and apply, while at the same time the preconditioned system (4.9) should be easy to solve. These two requirements are in competition with each other and it is necessary to find a balance between the two. For instance $\mathbf{P} = \mathbf{A}^{-1}$ is a perfect preconditioner in that just one iteration is needed, but $\mathbf{P}^{-1}$ is very time consuming to construct or apply, while on the other hand $\mathbf{P} = \mathbf{I}$ is constructed and applied in no time, but does not improve the convergence properties.

4.5.1 Incomplete LU factorisation

Since most systems are very large, we do not compute and store $\mathbf{P}^{-1}\mathbf{A}$ explicitly in general. It only is required that $\mathbf{P}\mathbf{w} = \mathbf{v}$ can be solved readily and hence needed that $\mathbf{w} = \mathbf{P}^{-1}\mathbf{v}$ can be computed easily for some arbitrary vector $\mathbf{v} \in \mathbb{C}^n$. The LU factorisation of $\mathbf{P}$ ensures that the preconditioner is easy to apply. However, the factorisation of a sparse matrix results in so-called fill-ins, that is, nonzero elements that emerge during the elimination process in positions that were initially equal to zero. The triangular factors $\mathbf{L}$ and $\mathbf{U}$ are therefore (far) less sparse than $\mathbf{A}$. This is one of the reasons why LU factorisation is not suitable for solving large sparse linear systems. However, by discarding part of the fill-in it turns out that we might obtain a good preconditioner $\mathbf{P} = \widetilde{\mathbf{L}}\widetilde{\mathbf{U}}$, where $\widetilde{\mathbf{L}}$ and $\widetilde{\mathbf{U}}$ are the approximate LU factors of $\mathbf{A}$.

The no-fill incomplete LU factorisation or ILU(0) factorisation was proposed in [15] by J.A. Meijerink and H.A. van der Vorst. It allows no fill in at all. For some systems the obtained preconditioner is very effective, but the no-fill factorisation is in general too crude an approximation of $\mathbf{A}$ and an iterative method might still require many iterations to converge. That is why more sophisticated incomplete LU factorisations are developed. These methods allow at least some fill-in

in the approximations $\widetilde{\mathbf{L}}$ and $\widetilde{\mathbf{U}}$.

The so-called ILUT(τ) or incomplete LU factorisation with a threshold or drop tolerance τ is proposed by Y. Saad and described in [20]. This method uses this drop tolerance τ as a dropping criterium: fill-ins are only allowed if their absolute value is greater than τ. Since the matrix can be poorly scaled, the drop tolerance is in general relative to (the norm of) the i-th row of $\mathbf{A}$: fill-in is accepted only if it is greater in absolute value than the drop tolerance $\tau\|\mathbf{A}_{(i,:)}\|$.

Chapter 5

Shifted Laplace preconditioning

Equations are more important to me, because politics is for the present,
but an equation is something for eternity.
Albert Einstein

We consider shifted Laplace preconditioning for the discrete Helmholtz equation (3.4) where the system matrix is equal to

$$\mathbf{A} = \mathbf{K} + ik\mathbf{C} - k^2\mathbf{M}. \tag{5.1}$$

The matrices $\mathbf{K}, \mathbf{C}$ and $\mathbf{M}$ are symmetric, real and psd and k is the acoustic wavenumber, which satisfies $k = 2\pi f/c_0$. We are in general interested in the solution to (3.4) for a range of frequencies $f_1, \ldots, f_n \in \mathbb{R}^+$, where the following preconditioner, based on [4], (with fixed $z \in \mathbb{C}$) is used:

$$\mathbf{P} = \mathbf{K} + ik\mathbf{C} - z\mathbf{M}. \tag{5.2}$$

We solve the preconditioned system (4.9) instead of the original system (4.1) and expect that the convergence properties improve. It is clear that choosing $z = k^2$ reduces the system to $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$, but we are only interested in preconditioners that are relatively cheap and applicable to a whole range of acoustic wavenumbers.

First, we consider the more general system matrix

$$\mathbf{A} = \mathbf{K} + ik\mathbf{C} - \ell\mathbf{M},$$

with $\ell = \ell(k) \in \mathbb{C}$ and $\mathrm{Re}(\ell) > 0$. We follow [7] and determine the eigenvalues λ of the preconditioned matrix

$$\mathbf{P}^{-1}\mathbf{A} = (\mathbf{K} + ik\mathbf{C} - z\mathbf{M})^{-1}(\mathbf{K} + ik\mathbf{C} - \ell\mathbf{M}),$$

that is, we solve

$$(\mathbf{K} + ik\mathbf{C} - z\mathbf{M})^{-1}(\mathbf{K} + ik\mathbf{C} - \ell\mathbf{M})\mathbf{v} = \lambda\mathbf{v}.$$

For $z = \ell$ we have that $\lambda = 1$. For $z \neq \ell$ we have that

$$(\mathbf{K} + ik\mathbf{C})\mathbf{v} = \frac{\lambda z - \ell}{\lambda - 1}\mathbf{M}\mathbf{v}.$$

We define $\mu = (\lambda z - \ell)/(\lambda - 1)$, which are the eigenvalues of the pencil $(\mathbf{K} + ik\mathbf{C}, \mathbf{M})$. Since z, λ and μ are in general complex, we write $z = z_r + iz_i$, $\lambda = \lambda_r + i\lambda_i$ and $\mu = \mu_r + i\mu_i$. It can be shown that $\mu_i \geq 0$ [7]. We obtain

$$\mu - \ell = \lambda(\mu - z),$$

and equivalently

$$\mu_r + i\mu_i - \ell = \lambda_r(\mu_r - z_r) + i\lambda_r(\mu_i - z_i) + i\lambda_i(\mu_r - z_r) + \lambda_i(-\mu_i + z_i),$$

and we split this equation in the real and complex parts

$$\begin{aligned} \mu_r - \ell &= \lambda_r(\mu_r - z_r) + \lambda_i(-\mu_i + z_i), \\ \mu_i &= \lambda_r(\mu_i - z_i) + \lambda_i(\mu_r - z_r). \end{aligned}$$

We distinguish between the three cases $z_i = 0$, $z_i < 0$ and $z_i > 0$.
If $z_i = 0$, the eigenvalues λ are in the half plane

$$(\ell - z)\lambda_i \geq 0. \tag{5.3}$$

This follows from the fact that for $z_i = 0$, the above equation for the complex part implies that for $\lambda_i \neq 0$

$$\mu_r = \mu_i \frac{1 - \lambda_r}{\lambda_i} + z,$$

and substitution of this equation into the equation of the real part results in

$$(\ell - z)\lambda_i = \mu_i(\lambda_i^2 + (\lambda_r - 1)^2).$$

Since $\mu_i \geq 0$, inequality (5.3) follows.

If $z_i < 0$, then the eigenvalues λ are inside or on the circle with centre c and radius r, where

$$c = \frac{\ell - \overline{z}}{z - \overline{z}}, \quad r = \left| \frac{z - \ell}{z - \overline{z}} \right|. \tag{5.4}$$

As was shown in [7], it suffices to prove that $|\lambda - c| \leq r$, which is done as follows:

$$\begin{aligned} |\lambda - c|^2 &= \left| \frac{\mu - \ell}{\mu - z} - \frac{\ell - \overline{z}}{z - \overline{z}} \right|^2 \\ &= \left| \frac{\mu(z - \ell) - (\ell - z)\overline{z}}{(\mu - z)(z - \overline{z})} \right|^2 \\ &= \left| \frac{\mu - \overline{z}}{\mu - z} \right|^2 r^2 \\ &= \frac{(\mu_r - z_r)^2 + (\mu_i + z_i)^2}{(\mu_r - z_r)^2 + (\mu_i - z_i)^2} r^2 \\ &\leq r^2. \end{aligned}$$

The inequality follows from the fact that $z_i < 0$ and $\mu_i \geq 0$.

If $z_i > 0$, then the eigenvalues λ are outside or on the circle with centre c and radius r. This can be shown analogous to the above.

5.1 Optimization of the shift

We restrict our analyses for the convergence properties to GMRES. An upper bound on the number of iterations of GMRES is given in (4.8). If all eigenvalues of $\mathbf{A}$ are enclosed in a circle centered at c and radius r (that does not enclose the origin), the convergence rate of GMRES is bounded by the inequality

$$\frac{\|\mathbf{r}_k\|}{\|\mathbf{r}_0\|} \leq \kappa(\mathbf{V}) \inf_{R_k \in \mathbb{P}_k} \left(\max_{\lambda \in \sigma(\mathbf{A})} |R_k(\lambda)| \right) \leq \kappa(\mathbf{V}) \left(\frac{r}{|c|} \right)^k, \tag{5.5}$$

where $\mathbf{V}$ is the matrix of eigenvectors of $\mathbf{A}$ and $\kappa(\mathbf{V}) = \|\mathbf{V}\| \, \|\mathbf{V}^{-1}\|$ is the spectral condition number of $\mathbf{V}$ in the 2-norm, which is typically small for Helmholtz problems. We refer to [19, p 866] for a proof of these inequalities. For $\ell \in \mathbb{R}$ we have that $|c| = r$ and to derive for this case an optimal shift, we consider first the more general case $\ell = \ell_r + i\ell_i \in \mathbb{C}$ with $r < |c|$ (which also has physical meaning, as ℓ with $\ell_i \leq 0$ comprises a damping coefficient for non-viscous fluids).

It is known that for multigrid methods the purely imaginary shift $z_r = 0$ works well [7]. The purely imaginary shift $z = iz_i$, which is shown to be optimal in [7] follows from minimizing the upper bound of (5.5), that is, by considering the minimum of the function

$$f(z_r, z_i) = \left(\frac{r}{|c|} \right)^2 = \frac{(z_r - \ell_r)^2 + (z_i - \ell_i)^2}{(z_r - \ell_r)^2 + (z_i + \ell_i)^2}.$$

The partial derivative

$$\frac{\partial f}{\partial z_i} = \frac{4\ell_i([z_i^2 - \ell_i^2] - [z_r - \ell_r]^2)}{([z_r - \ell_r]^2 + [z_i + \ell_i]^2)^2}$$

is equal to zero if $z_i^2 - \ell_i^2 - \ell_r^2 = 0$, so we choose $z = iz_i = \pm|\ell|i$. Since a shift with negative complex part results in a more favorable distribution of eigenvalues of $\mathbf{P}^{-1}\mathbf{A}$, as we have shown in the above, we choose $z_i \leq 0$ and hence set

$$z = -|\ell|i. \tag{5.6}$$

For $z_r \leq 0$ all eigenvalues of the preconditioner are in the right half-plane. We again choose $z_i \leq 0$ and since $\ell_r > 0$, we have that the partial derivative

$$\frac{\partial f}{\partial z_r} = \frac{8(z_r - \ell_r)z_i\ell_i}{([z_r - \ell_r]^2 + [z_i + \ell_i]^2)^2}$$

is negative for $z_r \leq 0$ and hence f takes its minimum on the edge $z_r = 0$. So, (5.6) is optimal for all choices of z with $z_r \leq 0$.

Because of continuity arguments, the choice (5.6) is still valid for the case $\ell_i = 0$ and $r = |c|$.

The linear systems that we consider can be written as (5.1) and hence we can confine ourselfs to the case $\ell = k^2$. We will consider two different choices for the shift of the shifted Laplace preconditioner (5.2), which we apply to sequences of linear systems (5.1) with $k = k_1, k_2, \ldots, k_n$. We consider the purely imaginary shift $z = -k_j^2 i$ (cf. (5.6)) and the real shift $z = k_j^2$, (with $j = 1, 2, \ldots, n$). For this last case, the preconditioned system matrix for the linear system with $k = k_j$ reduces to the identity matrix and obtaining the solution is trivial, but there is no analyses available on the performance of this preconditioner for $k \neq k_j$.

Chapter 6

Induced Dimension Reduction (s)

I don't need sleep, I need answers. I need to determine where in this swamp of unbalanced formulas squatteth the toad of truth.
Sheldon Cooper in *The Big Bang Theory* (Chuck Lorre, Bill Prady)

6.1 Induced Dimension Reduction

The method of Induced Dimension Reduction (IDR) for solving the system (4.1) with $\mathbf{A}$ a general square matrix was proposed by P. Sonneveld in 1980 [31] and is based on a three term recurrence for the residuals

$$\mathbf{r}_i = (\mathbf{I} - \omega_i \mathbf{A}) \left(\mathbf{r}_{i-1} - \frac{\mathbf{p} \cdot \mathbf{r}_{i-1}}{\mathbf{p} \cdot (\mathbf{r}_{i-1} - \mathbf{r}_{i-2})} (\mathbf{r}_{i-1} - \mathbf{r}_{i-2}) \right),$$

where $\mathbf{p} \in \mathbb{C}^n$ is some arbitrary vector, $\mathbf{r}_0$ an initial residual and $\mathbf{r}_1 = (\mathbf{I} - \omega_1 \mathbf{A})\mathbf{r}_0$. The above recurrence results almost always in $\mathbf{r}_{2n} = \mathbf{0}$, which means that $\mathbf{x}_{2n} = \mathbf{x}$. The residuals $\mathbf{r}_{2j}$ and $\mathbf{r}_{2j+1}$ live in the so-called Sonneveld spaces $\mathcal{G}_j$. These spaces are inductively defined as

$$\begin{aligned} \mathcal{G}_0 &= \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0), \\ \mathcal{G}_j &= (\mathbf{I} - \omega_j \mathbf{A})(\mathcal{G}_{j-1} \cap \mathcal{S}), \end{aligned} \tag{6.1}$$

where $\mathcal{S} = \text{span}\{\mathbf{p}\}^\perp$ and ω_j is a non-zero scalar. It can be shown that under mild conditions $\mathcal{G}_i \subset \mathcal{G}_{i-1}$ and $\mathcal{G}_k = \{\mathbf{0}\}$ for some $k \leq n$ [31, p. 551].

Given that $\mathbf{r}_{i-1}$ and $\mathbf{r}_i$ are in $\mathcal{G}_j$, we can compute a vector $\mathbf{s}_i = \mathbf{r}_i - \gamma_i \Delta\mathbf{r}_i \perp \mathbf{p}$ (with $\Delta\mathbf{r}_i = \mathbf{r}_i - \mathbf{r}_{i-1}$) by choosing $\gamma_i = -(\mathbf{p} \cdot \mathbf{r}_i)/(\mathbf{p} \cdot \Delta\mathbf{r}_i)$. Note that $\mathbf{s}_i \in \mathcal{G}_j \cap \mathcal{S}$ and it follows that

$$\mathbf{r}_{i+1} = (\mathbf{I} - \omega_j \mathbf{A})\mathbf{s}_i \in \mathcal{G}_{j+1}.$$

The scalar ω_j is fixed for two steps and chosen such that the convergence of the residuals improves. We set $\omega_j = (\mathbf{A}\mathbf{s}_i \cdot \mathbf{s}_i)/(\mathbf{A}\mathbf{s}_i \cdot \mathbf{A}\mathbf{s}_i)$ thus make sure that $\mathbf{r}_{i+1} \perp \mathbf{A}\mathbf{s}_i$ and hence that the norm of the $(i+1)$-th residual $\|\mathbf{r}_{i+1}\| = \|(\mathbf{I} - \omega\mathbf{A})\mathbf{s}\|$ is minimised with respect to ω.

Since $-\mathbf{A}\Delta\mathbf{x}_{i+1} = \Delta\mathbf{r}_{i+1} = (\mathbf{I} - \omega_j \mathbf{A})(\mathbf{r}_i - \gamma_i \Delta\mathbf{r}_i) - \mathbf{r}_i$ we can compute $\mathbf{x}_{i+1}$ by

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \omega_j \mathbf{r}_i - \gamma_i(\Delta\mathbf{x}_i + \omega_j \Delta\mathbf{r}_i).$$

The above results in the IDR algorithm as described in Algorithm 3.

6.2 Induced Dimension Reduction (s)

IDR(s) is a generalisation of IDR where instead of one hyperplane span$\{\mathbf{p}\}$ the intersection of $s \ll n$ distinct hyperplanes is used. We define

$$\mathcal{S} = \{\mathbf{p}_1, \ldots, \mathbf{p}_s\}^\perp,$$

Algorithm 3 Induced Dimension Reduction.

Require: $\mathbf{x}_0$, $\mathbf{p}$, tol $\in (0,1)$, maxit > 0
  $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\Delta\mathbf{g}_0 = \mathbf{0}$, $\Delta\mathbf{y}_0 = \mathbf{0}$, $\gamma = 0$, $i = 0$
  while $\|\mathbf{r}_i\| >$ tol and $i \leq$ maxit **do**
    $i = i + 1$
    $\mathbf{s} = \mathbf{r}_{i-1} + \gamma\Delta\mathbf{g}_{i-1}$
    $\mathbf{t} = \mathbf{A}\mathbf{s}$
    if $i = 1$ or i is even **then**
      $\omega_i = (\mathbf{t} \cdot \mathbf{s})/(\mathbf{t} \cdot \mathbf{t})$
    else
      $\omega_i = \omega_{i-1}$
    end if
    $\Delta\mathbf{x}_i = \gamma\Delta\mathbf{y}_{i-1} + \omega_i\mathbf{s}$
    $\Delta\mathbf{r}_i = \gamma\Delta\mathbf{g}_{i-1} - \omega_i\mathbf{t}$
    $\mathbf{x}_i = \mathbf{x}_{i-1} + \Delta\mathbf{x}_i$
    $\mathbf{r}_i = \mathbf{r}_{i-1} + \Delta\mathbf{r}_i$
    if i is even **then**
      $\Delta\mathbf{y}_i = \Delta\mathbf{y}_{i-1}$
      $\Delta\mathbf{g}_i = \Delta\mathbf{g}_{i-1}$
    else
      $\Delta\mathbf{y}_i = \Delta\mathbf{x}_i$
      $\Delta\mathbf{g}_i = \Delta\mathbf{r}_i$
    end if
    $\gamma = -(\mathbf{p} \cdot \mathbf{r}_i)/(\mathbf{p} \cdot \Delta\mathbf{g}_i)$
  end while

where the spaces $\mathcal{S}$ and $\mathcal{G}_0$ do not share a nontrivial invariant subspace of $\mathbf{A}$. This idea was conceived by P. Sonneveld and M.B. van Gijzen and described in [25] and [9]. The Sonneveld spaces $\mathcal{G}_j$ are again defined by (6.1) but we determine $s+1$ residuals in each space $\mathcal{G}_{j+1}$. Normally, the dimension of $\mathcal{G}_{j+1}$ equals the dimension of $\mathcal{G}_j$ minus s.

Given that $\mathbf{r}_{i-s}, \ldots, \mathbf{r}_i \in \mathcal{G}_j$, the residual $\mathbf{r}_{i+1}$ is forced to be in the space $\mathcal{G}_{j+1}$ by ensuring that

$$\mathbf{r}_{i+1} = (\mathbf{I} - \omega_{j+1}\mathbf{A})\mathbf{v}_i,$$

with $\mathbf{v}_i \in \mathcal{G}_j \cap \mathcal{S}$ and $\omega_{j+1} = (\mathbf{A}\mathbf{v}_i \cdot \mathbf{v}_i)/(\mathbf{A}\mathbf{v}_i \cdot \mathbf{A}\mathbf{v}_i)$ (so that the norm of this first residual in $\mathcal{G}_{j+1}$ is minimal). We ensure that the vector $\mathbf{v}_i$ is in $\mathcal{G}_j$ by choosing it equivalent to [25] as

$$\mathbf{v}_i = \mathbf{r}_i - \sum_{k=1}^{s} \gamma_k^i \Delta\mathbf{r}_{i-k}. \tag{6.2}$$

Orthonormalisation of the set of vectors $\{\mathbf{p}_1, \ldots, \mathbf{p}_s\}$ results in a set $\{\tilde{\mathbf{p}}_1, \ldots, \tilde{\mathbf{p}}_s\}$ and we define the matrix $\mathbf{P} = (\tilde{\mathbf{p}}_1, \ldots, \tilde{\mathbf{p}}_s)$. Now, $\mathbf{v}_i \in \mathcal{S}$ implies that it satisfies

$$\mathbf{P}^*\mathbf{v}_i = \mathbf{0}. \tag{6.3}$$

If we substitute (6.2) in (6.3), we obtain a set of s linear equations with s unknowns γ_k^i, which is almost always uniquely solvable. We define $\boldsymbol{\gamma}_i = (\gamma_1^i, \ldots, \gamma_s^i)^T$ and $\Delta\mathbf{R}_i = (\Delta\mathbf{r}_{i-1}, \ldots, \Delta\mathbf{r}_{i-s})$ and we can solve this linear system,

$$(\mathbf{P}^*\Delta\mathbf{R}_i)\boldsymbol{\gamma}_i = \mathbf{P}^*\mathbf{r}_i, \tag{6.4}$$

to obtain $\boldsymbol{\gamma}_i$ and compute the first residual $\mathbf{r}_{i+1} \in \mathcal{G}_{j+1}$ since we have that

$$\mathbf{r}_{i+1} = \mathbf{r}_i - \omega_{j+1}\mathbf{A}\mathbf{v}_i - \Delta\mathbf{R}_i\boldsymbol{\gamma}_i.$$

Other residuals $\mathbf{r}_{i+2}, \mathbf{r}_{i+3}, \ldots \in \mathcal{G}_{j+1}$ can be obtained by repeating the above calculations. We need (at least) $s+1$ residuals in $\mathcal{G}_{j+1}$ before we continue to the next space $\mathcal{G}_{j+2}$. The value ω_{j+1} is determined in the computation of $\mathbf{r}_{i+1}$, but is held fixed in the calculations of all subsequent residuals in $\mathcal{G}_{j+1}$. Note that the first $s+1$ residuals $\mathbf{r}_0, \ldots, \mathbf{r}_s$ should be in $\mathcal{G}_0 = \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)$ and since we generally have that $\mathcal{K}_n(\mathbf{A}, \mathbf{r}_0) = \mathbb{C}^n$, any set of $s+1$ (independent) vectors suffices.

After defining $\Delta\mathbf{X}_i = (\Delta\mathbf{x}_{i-1}, \ldots, \Delta\mathbf{x}_{i-s})$ we determine the approximate solutions $\mathbf{x}_i$ by

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \omega_{j+1}\mathbf{v}_i + \Delta\mathbf{X}_i\boldsymbol{\gamma}_i.$$

The choice (6.2) appears to be an unnecessary restriction and for larger values of s also leads to instability. Therefore, we follow [9] and note that since the residual differences $\Delta\mathbf{r}_i$ are in the space $\mathcal{G}_j$ any linear combination $\mathbf{g}_i$ of them is too. We obtain s of these vectors, after which we define the matrix $\mathbf{G}_i = (\mathbf{g}_{i-s}, \ldots, \mathbf{g}_{i-1})$. Since $\mathbf{g}_i \in \mathcal{G}_j$, there exist vectors $\mathbf{u}_i$ that satisfy $\mathbf{g}_i = \mathbf{A}\mathbf{u}_i$ and we define $\mathbf{U}_i = (\mathbf{u}_{i-s}, \ldots, \mathbf{u}_{i-1})$. By an equivalent reasoning to the above we obtain

$$\mathbf{v}_i = \mathbf{r}_i - \sum_{k=1}^{s} \gamma_k^i \mathbf{g}_{i-k} \tag{6.5}$$

and the recursions

$$\begin{aligned}\mathbf{r}_{i+1} &= \mathbf{r}_i - \omega_{j+1}\mathbf{A}\mathbf{v}_i - \mathbf{G}_i\boldsymbol{\gamma}_i \in \mathcal{G}_{j+1},\\ \mathbf{x}_{i+1} &= \mathbf{x}_i + \omega_{j+1}\mathbf{v}_i + \mathbf{U}_i\boldsymbol{\gamma}_i.\end{aligned}$$

The additional s residuals $\mathbf{r}_{i+2}, \ldots, \mathbf{r}_{i+s+1} \in \mathcal{G}_{j+1}$ and approximations $\mathbf{x}_{i+2}, \ldots, \mathbf{x}_{i+s+1}$ are determined by first computing the update vectors by

$$\mathbf{u}_{i+k} = \omega_{j+1}\mathbf{v}_{i+k} + \mathbf{U}_{k-1}\boldsymbol{\gamma}_{k-1} \quad \text{and} \quad \mathbf{g}_{i+k} = \mathbf{A}\mathbf{u}_{i+k}, \quad (k = 1, \ldots, s),$$

after which we compute

$$\mathbf{r}_{i+k+1} = \mathbf{r}_{i+k} - \mathbf{g}_{i+k} \quad \text{and} \quad \mathbf{x}_{i+k+1} = \mathbf{x}_{i+k} + \mathbf{u}_{i+k}, \quad (k = 1, \ldots, s).$$

The generalisation from residual differences $\Delta\mathbf{r}_i$ to vectors $\mathbf{g}_i \in \mathcal{G}_j$ allows us to choose the $\mathbf{g}_i$ in such a way that the algorithm improves. Choosing $\mathbf{g}_{i+k} \perp \mathbf{p}_j$ and $\mathbf{r}_{i+k+1} \perp \mathbf{p}_j$ ($j = 1, \ldots, k-1$ and $k = 1, \ldots, s$) ensures that the system (analogous to (6.4))

$$(\mathbf{P}^*\mathbf{G}_{i+k})\boldsymbol{\gamma}_{i+k} = \mathbf{P}^*\mathbf{r}_{i+k}$$

is easier to solve. This is because $\mathbf{g}_{i+k} \cdot \mathbf{p}_j = 0$ and $\mathbf{r}_{i+k+1} \cdot \mathbf{p}_j = 0$ and hence the system matrix $\mathbf{P}^*\mathbf{G}_{i+k}$ is lower triangular.

Since a small value of ω_{j+1} deteriorates the accuracy and convergence of IDR(s) (similar to such a negative effect on BiCGSTAB [23]), we increase ω_{j+1} if it is too small. To be more precise, we increase its value if the cosine of the angle between $\mathbf{A}\mathbf{v}_i$ and $\mathbf{v}_i$ is smaller than a value κ (where $\kappa = 0.7$ is recommended in [23]). The resulting IDR(s) algorithm is given in Algorithm 4.

For the sake of completeness, the Quasi Minimal Residual variant of the IDR algorithm, which applies the GMRES philosophy to the vectors $\mathbf{g}_i$, which can be found in Appendix A.

Algorithm 4 Induced Dimension Reduction (s).

Require: $\mathbf{x}_0$, $\mathbf{P}$, $s > 0$, tol $\in (0,1)$, maxit > 0
  $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\mathbf{g}_k = \mathbf{0}$, $\mathbf{y}_k = \mathbf{0}$ $(k = 1, \ldots, s)$, $\omega = 1$, $\kappa = 0.7$, $i = 0$
  while $\|\mathbf{r}_i\| >$ tol and $i \leq$ maxit **do**
    $i = i + 1$
    $\mathbf{f} = \mathbf{P}^* \mathbf{r}_{i-1}$, $(\phi_1, \ldots, \phi_s)^T = \mathbf{f}$
    for $j = 1, \ldots, s$ **do**
      solve $\mathbf{c}$ from $\mathbf{M}\mathbf{c} = \mathbf{f}$, $(\gamma_1, \ldots, \gamma_s)^T = \mathbf{c}$
      $\mathbf{v} = \mathbf{r}_{i-1} - \sum_{k=j}^{s} \gamma_k \mathbf{g}_k$
      $\mathbf{u}_i = \omega \mathbf{v} + \sum_{k=j}^{s} \gamma_k \mathbf{u}_k$
      $\mathbf{g}_j = \mathbf{A}\mathbf{u}_j$
      for $k = 1, \ldots, j-1$ **do**
        $\alpha = (\mathbf{p}_k \cdot \mathbf{g}_j)/\mu_{k,k}$
        $\mathbf{g}_j = \mathbf{g}_j - \alpha \mathbf{g}_k$
        $\mathbf{u}_j = \mathbf{u}_j - \alpha \mathbf{u}_k$
      end for
      $\mu_{k,j} = \mathbf{p}_k \cdot \mathbf{g}_j$, $\mathbf{M}_{k,j} = \mu_{k,j}$ $(k = j, \ldots, s)$
      $\beta = \phi_j / \mu_{j,j}$
      $\mathbf{r}_{i-1} = \mathbf{r}_{i-1} - \beta \mathbf{g}_j$
      $\mathbf{x}_{i-1} = \mathbf{x}_{i-1} + \beta \mathbf{u}_j$
      if $j + 1 \leq s$ **then**
        $\phi_k = 0$ $(k = 1, \ldots, j)$
        $\phi_k = \phi_k - \beta \mu_{k,j}$ $(k = j+1, \ldots, s)$
        $\mathbf{f} = (\phi_1, \ldots, \phi_s)^T$
      end if
    end for
    $\mathbf{t} = \mathbf{A}\mathbf{r}_{i-1}$
    $\omega = (\mathbf{t} \cdot \mathbf{r}_{i-1})/(\mathbf{t} \cdot \mathbf{t})$
    $\rho = (\mathbf{t} \cdot \mathbf{r}_{i-1})/(\|\mathbf{t}\| \|\mathbf{r}_{i-1}\|)$
    if $|\rho| < \kappa$ **then**
      $\omega = \omega \kappa / |\rho|$
    end if
    $\mathbf{r}_i = \mathbf{r}_{i-1} - \omega \mathbf{t}$
    $\mathbf{x}_i = \mathbf{x}_{i-1} + \omega \mathbf{v}$
  end while

6.3 The initial search space $\mathcal{U}_0$

The IDR(s) method constructs $s + 1$ vectors $\mathbf{g}_i$ in each space $\mathcal{G}_j$. For $j > 0$, these vectors are based on the $s + 1$ vectors in $\mathcal{G}_{j-1}$, while the $s + 1$ vectors in $\mathcal{G}_0$ are obtained by initialising the s vectors as $\mathbf{g}_i = \mathbf{0}$, after which s iterations are carried out. Since each vector $\mathbf{g}_i$ corresponds to a vector $\mathbf{u}_i$ through $\mathbf{g}_i = \mathbf{A}\mathbf{u}_i$, we can choose s vectors to be in the so-called initial search space $\mathcal{U}_0$ and in this way fill the space $\mathcal{G}_0$ with $s + 1$ vectors. Since $\mathcal{G}_0 = \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)$ and $\mathcal{K}_n(\mathbf{A}, \mathbf{r}_0) = \mathbb{C}^n$ under mild conditions, any n-dimensional vector suffices.

Approximations to eigenvectors of the system matrix $\mathbf{A}$ are good candidates for this initial search space $\mathcal{U}_0$, but for the given problem these vectors cannot be obtained easily. Since the approximate solutions $\mathbf{x}_i$ lie in the space $\mathcal{U}_j$, we could for a sequence of Helmholtz problems use the solutions

to s problems that most likely have some agreement to the solution of the problem that we are about to solve.

6.4 Eigenvalue approximation with IDR(s)

Since IDR(s) is a Krylov subspace method, the approximations $\mathbf{x}_i$ and residuals $\mathbf{r}_i$ are based on a polynomial in $\mathbf{A}$, as we showed in section 4.2. The residuals $\mathbf{r}_i$ satisfy

$$\mathbf{r}_i = R_i(\mathbf{A})\mathbf{r}_0 = [\mathbf{I} - \mathbf{A}P_{i-1}(\mathbf{A})]\mathbf{r}_0,$$

with $R_i(0) = 1$ and $P_{i-1}(\mathbf{A})$ a polynomial of degree $i-1$, such that $\mathbf{x}_i = \mathbf{x}_0 + P_{i-1}(\mathbf{A})\mathbf{r}_0 \in \mathcal{K}_i(\mathbf{A}, \mathbf{r}_0)$. This implies that IDR($s$), and in fact any Krylov subspace method, can be seen as a method that constructs a residual polynomial R_i in $\mathbf{A}$.

Since we have that $\mathbf{r}_i \in \mathcal{G}_j$, we can write the residuals as

$$\mathbf{r}_i = \prod_{\ell=1}^{j}(\mathbf{I} - \omega_\ell \mathbf{A})\hat{\mathbf{r}}_i = \Omega_j(\mathbf{A})\hat{\mathbf{r}}_i \tag{6.6}$$

for certain vectors $\hat{\mathbf{r}}_i$, where $\Omega_j(\xi) = (1-\omega_1\xi)\cdot(1-\omega_2\xi)\cdots(1-\omega_j\xi)$. The roots of this polynomial are $\xi_k = 1/\omega_k$ $(k = 1, \ldots, j)$.

Note that the $\hat{\mathbf{r}}_i$ are in the Sonneveld space $\mathcal{G}_0$, and hence $\hat{\mathbf{r}}_i$ can be written as

$$\hat{\mathbf{r}}_i = \Psi_{i-j}(\mathbf{A})\mathbf{r}_0,$$

where Ψ_{i-j} is some polynomial of degree $i-j$. This implies that the residual polynomial equals

$$R_i(\xi) = \Omega_j(\xi)\Psi_{i-j}(\xi).$$

Due to finite terminiation, the roots of $\Psi_{i-j}(\xi)$ should converge to the eigenvalues, see [9]. The residual polynomial $R_i(\mathbf{A})$ is optimal if it is minimal on the spectrum of $\mathbf{A}$, which follows from the estimate $\|\mathbf{r}_i\| \leq \|R_i(\mathbf{A})\|\|\mathbf{r}_0\|$. The value ω_j is traditionally determined by the minimisation of the norm of the first residual $\mathbf{r}_i$ with respect to ω_j. However, if bounds on the spectrum $\sigma(\mathbf{A})$ are known, we can base our choice for the values ω_j on this information, which might lead to a polynomial that is smaller on the spectrum and hence might result in a reduction of the number of iterations that are needed for convergence.

If the system matrix is explicitly available (for instance, if we do not apply a preconditioner) several bounds on the spectrum can be determined relatively easily, see [29]. Any matrix can for example be written as the sum of two Hermitian matrices, as follows:

$$\mathbf{A} = \frac{1}{2}(\mathbf{A} + \mathbf{A}^*) + i\frac{1}{2i}(\mathbf{A} - \mathbf{A}^*) = \Re(\mathbf{A}) + i\Im(\mathbf{A}),$$

with $\Re(\mathbf{A}) = \frac{1}{2}(\mathbf{A} + \mathbf{A}^*)$ and $\Im(\mathbf{A}) = \frac{1}{2i}(\mathbf{A} + \mathbf{A}^*)$. Since an eigenvalue λ_i of any square matrix $\mathbf{A}$ lies within at least one of the Gershgorin discs $D(a_{ii}, R_i)$, where $R_i = \sum_{j\neq i}|a_{ij}|$, we can determine a bounding box by applying the Gershgorins circle theorem to both $\Re(\mathbf{A})$ and $\Im(\mathbf{A})$. In other words, we obtain the following bounds on the eigenvalues: $m^r \leq \mathrm{Re}(\lambda_i) \leq M^r$ and $m^i \leq \mathrm{Im}(\lambda_i) \leq M^i$.

Since the system matrix is often not explicitly available, we need other ways to determine a certain set in which the spectrum of the matrix is contained. For large matrices, this is generally very difficult and we therefore use IDR(s) to compute (an approximation of) the eigenvalues of the system matrix. The obtained eigenvalue approximations are known as Ritz values. The set of all possible Ritz values is the so-called numerical range or field of values, which is defined as

$$\mathrm{FOV}(\mathbf{A}) = \left\{ \frac{\mathbf{x} \cdot \mathbf{A}\mathbf{x}}{\mathbf{x} \cdot \mathbf{x}} \ : \ \mathbf{x} \in \mathbb{C}^n \setminus \{\mathbf{0}\} \right\}. \tag{6.7}$$

In GMRES, the Arnoldi iteration of section 4.3, which was primarily seen as a way to reduce a matrix $\mathbf{A}$ to an upper Hessenberg matrix, generates an orthonormal basis for the Krylov subspace and constructs (partial) Hessenberg factorisations. The eigenvalues of these matrices, the Ritz values, are accurate eigenvalue approximations. The Arnoldi relation (4.4) can be written as

$$\mathbf{A}\mathbf{Q}_j = \mathbf{Q}_j\mathbf{H}_j + h_{j+1,j}\mathbf{q}_{j+1}\mathbf{e}_j^*, \tag{6.8}$$

from which it follows that if $h_{j,j-1} = 0$, the columns of $\mathbf{Q}_{j-1}$ span an eigenspace of $\mathbf{A}$. If $h_{j,j-1} \neq 0$, the columns approximate an eigenspace of $\mathbf{A}$ and the eigenvalues of $\mathbf{H}_{j-1}$ might be good approximations of the eigenvalues of $\mathbf{A}$.

6.4.1 Derivation of the eigenvalue approximation

It is possible to use IDR(s) to derive an equation similar to (4.4), but in this case, an equivalent of the matrix $\mathbf{Q}_{j-1}$ is not explicitly available. We are, however, still able to obtain the matrix $\mathbf{H}_i$ and the desired Ritz values.

To simplify the analyses, we introduce a new way of indexing, that is, we consider the residuals $\mathbf{r}_k^j = \mathbf{r}_i$, where the relation between these indices is given by $i = (j-1)s + k$. This means that the residuals $\mathbf{r}_k^j$ $(k = 0, \ldots, s)$ are in the space $\mathcal{G}_j$.

The first vector $\mathbf{r}_0^{j+1}$ in the new space $\mathcal{G}_{j+1}$ is given by

$$\mathbf{r}_0^{j+1} = (\mathbf{I} - \omega_{j+1}\mathbf{A})\mathbf{r}_s^j,$$

and all residuals $\mathbf{r}_k^{j+1} \in \mathcal{G}_{j+1}$ $(k = 1, \ldots, s)$ are obtained by the use of vectors $\mathbf{v}_k^j$ and $\mathbf{g}_k^{j+1}$, as follows:

$$\begin{aligned}
\mathbf{v}_k^j &= \mathbf{r}_{k-1}^{j+1} - \sum_{\ell=k}^{s} \gamma_{\ell,k}^j \mathbf{g}_\ell^j \\
\mathbf{g}_k^{j+1} &= \mathbf{r}_{k-1}^{j+1} - (\mathbf{I} - \omega_{j+1}\mathbf{A})\mathbf{v}_k^j - \sum_{\ell=1}^{k-1} \alpha_{\ell,k}^{j+1} \mathbf{g}_\ell^{j+1} \\
\mathbf{r}_k^{j+1} &= \mathbf{r}_{k-1}^{j+1} - \beta_k^{j+1}\mathbf{g}_k^{j+1}
\end{aligned}$$

We substitute the third and first equation in the second and obtain

$$\begin{aligned}\frac{1}{\beta_k^{j+1}}(\mathbf{r}_{k-1}^{j+1}-\mathbf{r}_k^{j+1}) &= \mathbf{r}_{k-1}^{j+1}-(\mathbf{I}-\omega_{j+1}\mathbf{A})\mathbf{v}_k^j-\sum_{\ell=1}^{k-1}\frac{\alpha_{\ell,k}^{j+1}}{\beta_\ell^{j+1}}(\mathbf{r}_{\ell-1}^{j+1}-\mathbf{r}_\ell^{j+1})\\ &= \mathbf{r}_{k-1}^{j+1}-(\mathbf{I}-\omega_{j+1}\mathbf{A})\left(\mathbf{r}_{k-1}^{j+1}-\sum_{\ell=k}^{s}\gamma_{\ell,k}^j\mathbf{g}_\ell^j\right)-\sum_{\ell=1}^{k-1}\frac{\alpha_{\ell,k}^{j+1}}{\beta_i^{j+1}}(\mathbf{r}_{\ell-1}^{j+1}-\mathbf{r}_i^{j+1})\\ &= \mathbf{r}_{k-1}^{j+1}-(\mathbf{I}-\omega_{j+1}\mathbf{A})\left(\mathbf{r}_{k-1}^{j+1}-\sum_{\ell=k}^{s}\frac{\gamma_{\ell,k}^j}{\beta_\ell^j}(\mathbf{r}_{\ell-1}^j-\mathbf{r}_\ell^j)\right)-\sum_{\ell=1}^{k-1}\frac{\alpha_{\ell,k}^{j+1}}{\beta_\ell^{j+1}}(\mathbf{r}_{\ell-1}^{j+1}-\mathbf{r}_\ell^{j+1})\end{aligned}$$

and we can write this equivalently as

$$\mathbf{A}\mathbf{r}_{k-1}^{j+1}=\frac{1}{\omega_{j+1}}\left(-(\mathbf{I}-\omega_{j+1}\mathbf{A})\sum_{\ell=k}^{s}\frac{\gamma_{\ell,k}^j}{\beta_\ell^j}(\mathbf{r}_{\ell-1}^j-\mathbf{r}_\ell^j)+\sum_{\ell=1}^{k}\frac{\alpha_{\ell,k}^{j+1}}{\beta_\ell^{j+1}}(\mathbf{r}_{\ell-1}^{j+1}-\mathbf{r}_\ell^{j+1})\right), \tag{6.9}$$

with $\alpha_{k,k}^{j+1}=1$.

Each zeroth residual $\mathbf{r}_0^{j+1}$ satisfies $\mathbf{r}_0^{j+1}=(\mathbf{I}-\omega_{j+1}\mathbf{A})\mathbf{r}_s^j$, and hence it follows from (6.6) that

$$\hat{\mathbf{r}}_0^{j+1}=\hat{\mathbf{r}}_s^j. \tag{6.10}$$

For $k=1,\ldots,s$, we substitute (6.6) into (6.9) and cancel the product terms, which results in the relation

$$\mathbf{A}\hat{\mathbf{r}}_{k-1}^{j+1}=\frac{1}{\omega_{j+1}}\left(-\sum_{\ell=k}^{s}\frac{\gamma_{\ell,k}^j}{\beta_\ell^j}(\hat{\mathbf{r}}_{\ell-1}^j-\hat{\mathbf{r}}_\ell^j)+\sum_{\ell=1}^{k}\frac{\alpha_{\ell,k}^{j+1}}{\beta_\ell^{j+1}}(\hat{\mathbf{r}}_{\ell-1}^{j+1}-\hat{\mathbf{r}}_\ell^{j+1})\right).$$

We rewrite this as

$$\mathbf{A}\hat{\mathbf{r}}_{k-1}^{j+1}=\frac{1}{\omega_{j+1}}\left(-\sum_{\ell=k-1}^{s-1}\frac{\gamma_{\ell+1,k}^j}{\beta_{\ell+1}^j}\hat{\mathbf{r}}_\ell^j-\sum_{\ell=k}^{s}\frac{\gamma_{\ell,k}^j}{\beta_\ell^j}\hat{\mathbf{r}}_\ell^j+\sum_{\ell=0}^{k-1}\frac{\alpha_{\ell+1,k}^{j+1}}{\beta_{\ell+1}^{j+1}}\hat{\mathbf{r}}_\ell^{j+1}-\sum_{\ell=1}^{k}\frac{\alpha_{\ell,k}^{j+1}}{\beta_\ell^{j+1}}\hat{\mathbf{r}}_\ell^{j+1}\right),$$

and after recombining the summation terms and substitution of (6.10), we end up with

$$\begin{aligned}\mathbf{A}\hat{\mathbf{r}}_{k-1}^{j+1}=\frac{1}{\omega_{j+1}}\Bigg(&-\frac{\gamma_{k,k}^j}{\beta_k^j}\hat{\mathbf{r}}_{k-1}^j-\sum_{\ell=k}^{s-1}\left[\frac{\gamma_{\ell+1,k}^j}{\beta_{\ell+1}^j}-\frac{\gamma_{\ell,k}^j}{\beta_\ell^j}\right]\hat{\mathbf{r}}_\ell^j+\left[\frac{\alpha_{1,k}^{j+1}}{\beta_1^{j+1}}+\frac{\gamma_{s,k}^j}{\beta_s^j}\right]\hat{\mathbf{r}}_s^j\\ &+\sum_{\ell=1}^{k-1}\left[\frac{\alpha_{\ell+1,k}^{j+1}}{\beta_{\ell+1}^{j+1}}-\frac{\alpha_{\ell,k}^{j+1}}{\beta_\ell^{j+1}}\right]\hat{\mathbf{r}}_\ell^j-\frac{\alpha_{k,k}^{j+1}}{\beta_k^{j+1}}\hat{\mathbf{r}}_k^{j+1}\Bigg).\end{aligned}$$

By choosing appropriate values $h_{\ell,k}^j$ we can simplify this to

$$\mathbf{A}\hat{\mathbf{r}}_{k-1}^{j+1}=\sum_{\ell=k-1}^{s}h_{\ell,k}^j\hat{\mathbf{r}}_\ell^j+\sum_{\ell=1}^{k}h_{\ell,k}^{j+1}\hat{\mathbf{r}}_\ell^{j+1},$$

and if we use the original indices it reduces to

$$\mathbf{A}\hat{\mathbf{r}}_{i-1}=\sum_{\ell=i-s-1}^{i}h_\ell\hat{\mathbf{r}}_\ell.$$

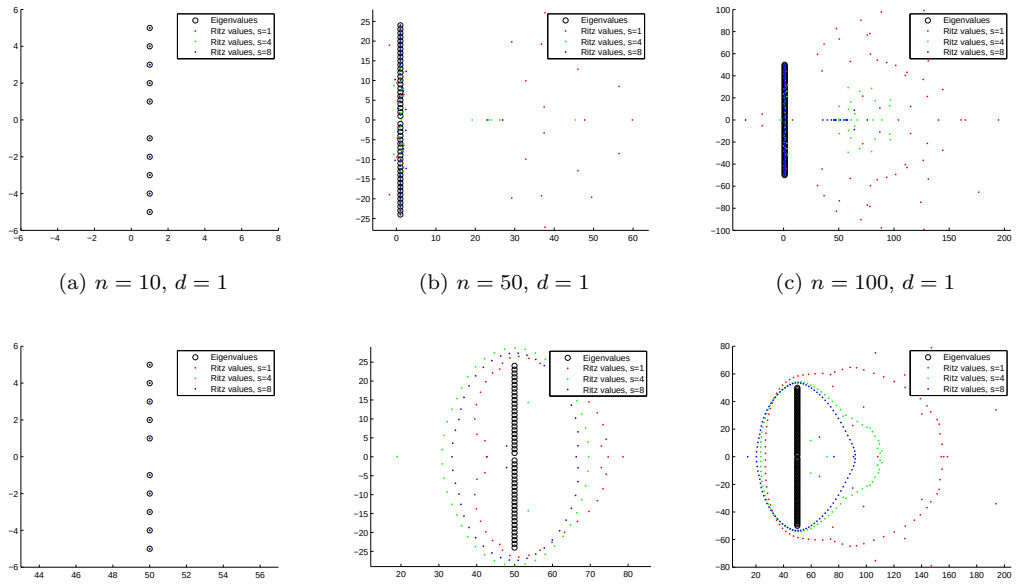
This leads to the relation

$$\mathbf{A}\hat{\mathbf{R}}_i = \hat{\mathbf{R}}_i \mathbf{H}_i + h_{i+1,i}\mathbf{r}_{i+1}\mathbf{e}_i^*, \tag{6.11}$$

with $\hat{\mathbf{R}}_i = [\hat{\mathbf{r}}_0\ \hat{\mathbf{r}}_1 \cdots\ \hat{\mathbf{r}}_i]$ and $\underline{\mathbf{H}}_i$ an $(i+1) \times i$ extended Hessenberg matrix with upper bandwidth s and lower bandwidth 1 (and hence $s+2$ non-zero diagonals). This equation is similar to the Arnoldi relation (6.8), but in this case $\tilde{\mathbf{R}}_i$ is not known explicitly and it is not a matrix with orthonormal columns. The elements of $\mathbf{H}_i$ are the values h_i, which depend on the coefficients ω_j and $\alpha_{\ell,i}$, $\beta_{\ell,i}$ and $\gamma_{\ell,i}$. Its eigenvalues are an approximation to (some of) the eigenvalues of $\mathbf{A}$.

6.4.2 Validation and tests

We validate the above analyses by considering a few test matrices of which the precise spectrum can be computed easily. We use several matrices that are included in the Matlab gallery of test matrices, which is based upon the work of N. J. Higham, see [11]. First we use as system matrix the Hanowa matrix. This sparse $n \times n$ matrix has non-zero entries $a_{i,i} = d$ and $a_{i,i+n} = i = a_{i+n,i}$. The eigenvalues of this matrix are $d \pm ik$ with $k = 1, 2, \ldots n/2$. We perform $n + \lfloor n/s \rfloor$ iterations, which results in the matrix $\mathbf{H}_n$ after which the eigenvalues of this matrix, the Ritz values, are computed.



(a) $n = 10$, $d = 1$ (b) $n = 50$, $d = 1$ (c) $n = 100$, $d = 1$

(d) $n = 10$, $d = 50$ (e) $n = 50$, $d = 50$ (f) $n = 100$, $d = 50$

Figure 6.1: Eigenvalues and Ritz values of the Hanowa matrix of size $n \times n$.

The distribution of the eigenvalues, indicated with a black 'o', and the Ritz values, specified with red (for $s = 1$), green (for $s = 4$) or blue (for $s = 8$) dots, are displayed in Figure 6.1. We see that for small Hanowa matrices, the Ritz values do converge to all the eigenvalues of $\mathbf{A}$. Increasing s leads to more accurate Ritz values for the larger matrices and we indeed observe that the exterior eigenvalues are better approximated by the Ritz values.

For increasing matrix size n, some of the Ritz values are located farther away from the actual spectrum of $\mathbf{A}$. Similar behaviour has been described by M.H. Gutknecht and J.M. Zemke in [10].

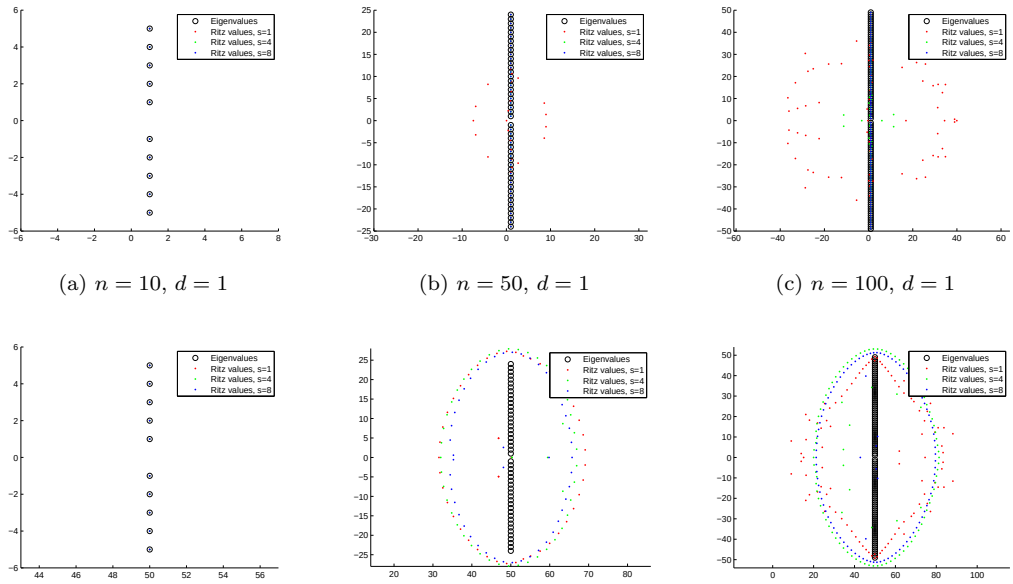
They suggest that this is caused by the fact that the reciprocals of ω_j are outside of the set (6.7), the field of values of $\mathbf{A}$.

In [22], V. Simoncini and D.B. Szyld consider a case where the reciprocals of ω_j are in this field of values. They choose for ω_j^{-1} the Ritz values that result from a preliminary generation of a small Krylov subspace.

Alternatively, we consider

$$\omega_j = \frac{\mathbf{v}_i \cdot \mathbf{v}_i}{\mathbf{v}_i \cdot \mathbf{A}\mathbf{v}_i},$$

and thus also make sure that $\omega_j^{-1} \in \mathrm{FOV}(\mathbf{A})$. This choice of ω_j is known as the method of steepest descent and results from choosing ω_j such that $(\mathbf{I} - \omega_j \mathbf{A})\mathbf{v}_i \perp \mathbf{v}_i$. If we repeat the above test with this choice of ω_j, the results are as in Figure 6.2.



(a) $n = 10$, $d = 1$ (b) $n = 50$, $d = 1$ (c) $n = 100$, $d = 1$

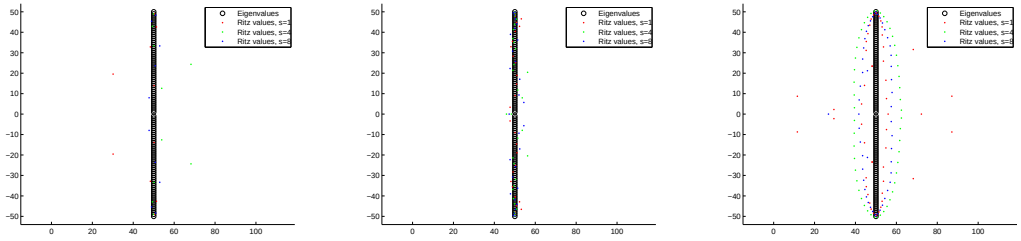
(d) $n = 10$, $d = 50$ (e) $n = 50$, $d = 50$ (f) $n = 100$, $d = 50$

Figure 6.2: Eigenvalues and Ritz values of the Hanowa matrix of size $n \times n$.

The resulting Ritz values resemble the eigenvalues more accurately, and the Ritz values that are still distant from the spectrum are more close to the spectrum compared to the standard choice of ω_j. The Ritz values are also more evenly distributed around the spectrum. Therefore, we choose the method of steepest descent for ω_j in all upcoming tests.

We note that the Ritz values that are not accurately approximating eigenvalues all lie on some kind of oval close to the line on which these eigenvalues are located, see for instance Figure 6.2(e)-(f). If we consider only $\mathbf{H}_k$ (which corresponds to taking only $k + \lfloor k/s \rfloor$ iterations for some $k < n$), this oval is less apparent and the Ritz values show more resemblance with the eigenvalues. In Figure 6.3 we display the results for $k = 10, 30, 50$ for the Hanowa matrix with $n = 100$ and $d = 50$. For increasing s, the minimal surface that encloses this oval is smaller and hence the cooresponding

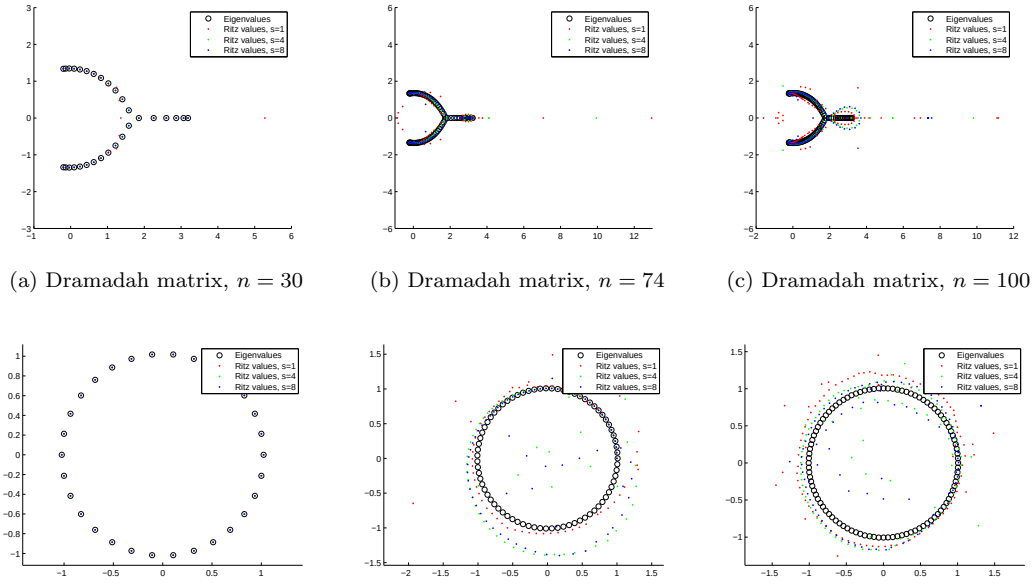
Ritz values are a better approximation to the eigenvalues.



(a) $k = 10$ (b) $k = 30$ (c) $k = 50$

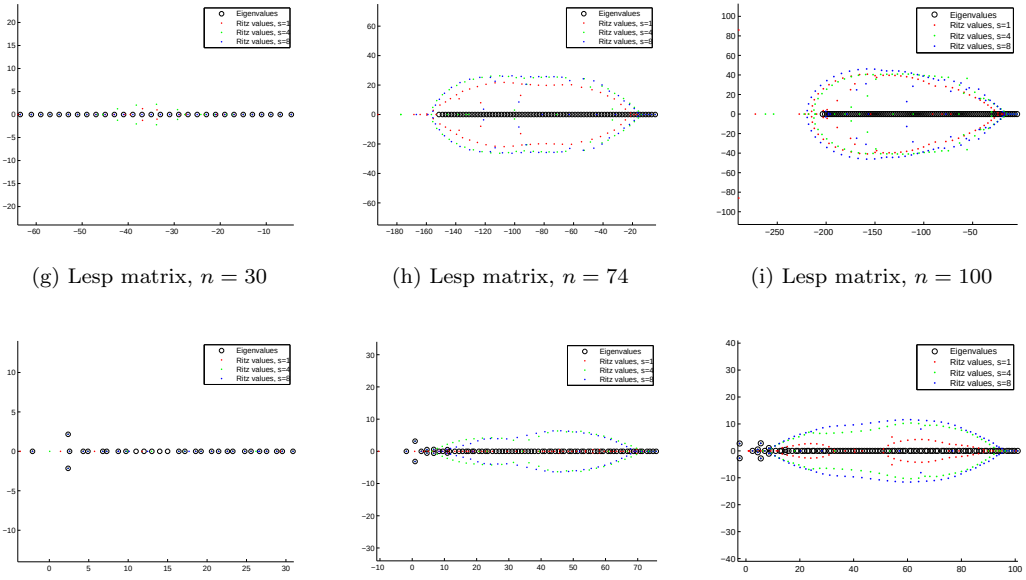
Figure 6.3: Eigenvalues of the Hanowa matrix of size 100×100 and Ritz values based on $\mathbf{H}_k$.

We consider a few other simple matrices of which the spectrum has a particular shape, and compare for $s = 1, 4, 8$ the Ritz values with the eigenvalues of these matrices, see Figure 6.4. The Dramadah matrix has entries equal to 1 on the first superdiagonal, the main diagonal and all even subdiagonals and its spectrum can be described as a rotated Y-shape. The Smoke matrix is a sparse matrix with the n roots of unity on the main diagonal and the non-zero entries $a_{i,i+1} = 1 = a_{n,1}$. The eigenvalues of this matrix all lie on the unit circle. The Lesp matrix is a tridiagonal matrix with the entries on the diagonal $a_{i,i} = -2i - 3$, on the superdiagonal $a_{i-1,i} = i$ and the subdiagonal $a_{i,i-1} = 1/i$. Its eigenvalues are distributed evenly on the interval $[-2n-3.5, -4.5]$. The Riemann matrix is a full matrix with $a_{i,j} = i$ if $i+1 \mid j+1$ and $a_{i,j} = -1$ otherwise. Almost all eigenvalues are real and positive and the integers on $(n/3, n/2]$ are eigenvalues.



(a) Dramadah matrix, $n = 30$ (b) Dramadah matrix, $n = 74$ (c) Dramadah matrix, $n = 100$

(d) Smoke matrix, $n = 30$ (e) Smoke matrix, $n = 75$ (f) Smoke matrix, $n = 100$



(g) Lesp matrix, $n = 30$ (h) Lesp matrix, $n = 74$ (i) Lesp matrix, $n = 100$

(j) Riemann matrix, $n = 30$ (k) Riemann matrix, $n = 75$ (l) Riemann matrix, $n = 100$

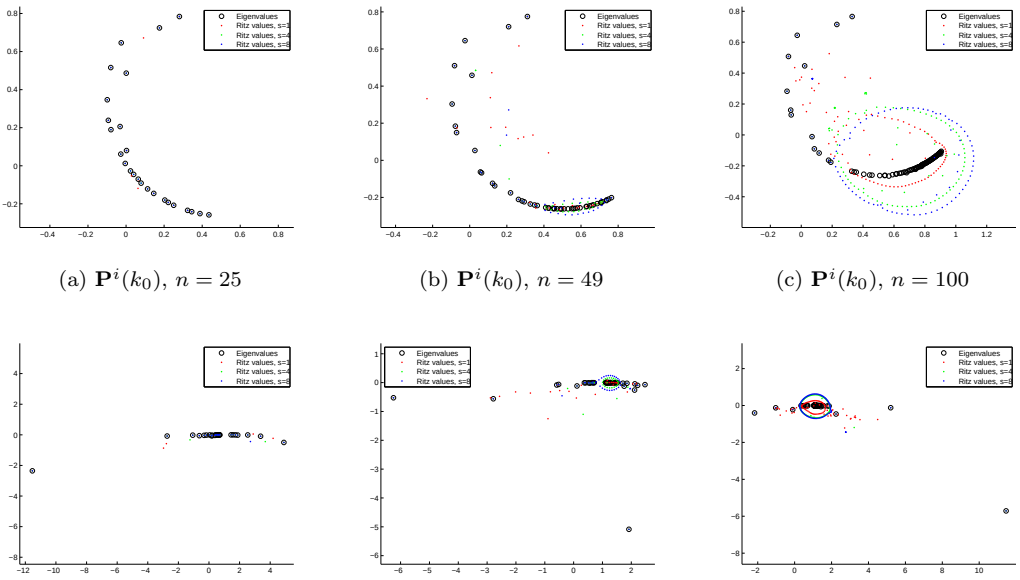
Figure 6.4: Eigenvalues and Ritz values of several matrices of size $n \times n$.

For rather small sized matrices (i.e. $n = 30$), the Ritz values almost always coincide with the eigenvalues of the system matrix, especially for the case $s = 8$. If the matrix gets larger, the Ritz values are again less accurate. In a few cases, a single Ritz value is very distant from the spectrum. This can happen for any value of s and any value of n. If we consider for instance the Riemann matrix, we see in Figure 6.4(k) that with $n = 75$ all Ritz values are relatively close to the spectrum, while for $n = 74$ and $s = 8$, we obtain a single Ritz value close to 1952 and with $n = 76$ and $s = 1$ a Ritz value is close to -480. All other Ritz values in these two cases are just as close to the spectrum as for the case $n = 75$.

In the last validation test, we focus on the room problem of section 3.1 with the system matrix (3.7), which depends on the acoustic wavenumber k. We consider the two shifted Laplace preconditioners that are described in section 5.1. The purely imaginary shifted preconditioner $\mathbf{P}^i(k_0) = \mathbf{K} + ik_0\mathbf{C} + ik_0^2\mathbf{M}$ has as shift $-ik_0^2$ and the preconditioner with a real shift k_0^2 equals $\mathbf{P}^r(k_0) = \mathbf{K} + ik_0\mathbf{C} - k_0^2\mathbf{M}$. The resulting preconditioned system matrices $[\mathbf{P}^*(k_0)]^{-1}\mathbf{A}(k)$ are used in the experiments of the Literature Review, see Appendix B. In this test, we set $k_0 = 120 \cdot 2\pi/c_0$ and $k = 100 \cdot 2\pi/c_0$. For a small number of unknowns n, we are able to compute the eigenvalues of the preconditioned system matrix and take enough iterations to construct the Hessenberg matrix of size $n \times n$, which leads to n Ritz values. The results are displayed in Figure 6.5.

The exterior Ritz values match with the exterior eigenvalues of the system matrix, while the more clustered eigenvalues are not approximated very well. We also observe that for $n = 49$ and $n = 100$ the oval around (a part of) the spectrum is again present. It is remarkable that this oval is worse for increasing s.

We are of course interested in an approximation of the eigenvalues of a large system matrix of size



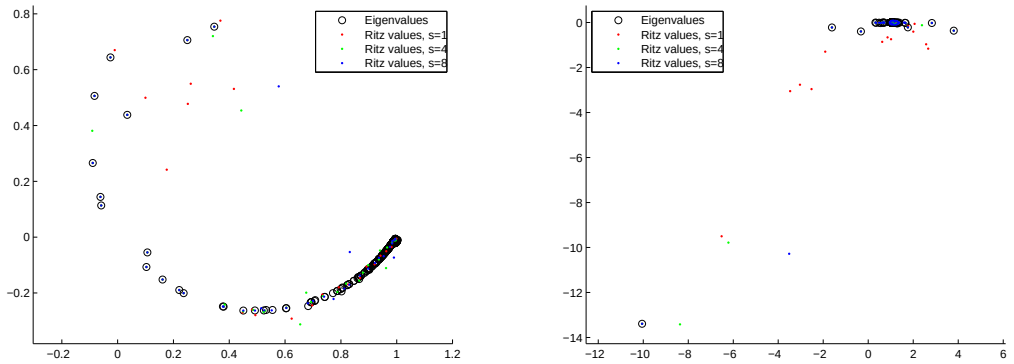
(a) $\mathbf{P}^i(k_0)$, $n = 25$ (b) $\mathbf{P}^i(k_0)$, $n = 49$ (c) $\mathbf{P}^i(k_0)$, $n = 100$

(d) $\mathbf{P}^r(k_0)$, $n = 25$ (e) $\mathbf{P}^r(k_0)$, $n = 49$ (f) $\mathbf{P}^r(k_0)$, $n = 100$

Figure 6.5: Validation on the room problem with the $n \times n$ system matrix $[\mathbf{P}^*(k_0)]^{-1}\mathbf{A}(k)$.

$n \times n$ without the need of n iterations. Taking less than n iterations was very effective for the above test matrices, which we motivated by Figure 6.3, not only for reasons concerning computation time, but also for a better approximation to the eigenvalues. We consider the room problem with $n = 1600$ equations and consider $k+\lfloor k/s \rfloor$ iterations with $k = 40$, see Figure 6.6 for the results.

We see that some inaccurate Ritz values are present for all values of s, but the distribution of the eigenvalues is very accurately represented by the 40 Ritz values.



(a) $\mathbf{P}^i(k_0)$ (b) $\mathbf{P}^r(k_0)$

Figure 6.6: Validation on the room problem with $n = 1600$ and $k = 40$ Ritz values.

The above tests validate the analyses of the preceding section. For very small matrices, all the eigenvalues can be found by computing n Ritz values. For larger matrices, this does not work. Since we are in general not interested in computing all n eigenvalues, this is not an obstacle.

The lack of accuracy is most probably caused by numerical instability and sensitivity to rounding errors. The current implementation follows directly from the analysis and the eigenvalue estimation is just a by-product of IDR(s).

6.4.3 The Chebyshev polynomial

The Ritz values tend to approximate the exterior eigenvalues better than the eigenvalues in the interior part of the spectrum, see [2, Ch 3.2]. This implies that if we construct a convex set $\mathcal{C} \subset \mathbb{C}$ that encloses the Ritz values, the spectrum of $\mathbf{A}$ is most probably contained in this set. If we then construct a polynomial that is small on this set, it is also small on the spectrum.

A promising candidate is a polynomial based on the so-called Chebyshev polynomial $T_m^{\mathcal{C}}(\xi)$, This polynomial is monic and of degree m and is defined to be minimal on a set $\mathcal{C}$, that is, it satisfies

$$\|T_m^{\mathcal{C}}(\xi)\|_\infty \leq \|P^{\mathcal{C}}(\xi)\|_\infty,$$

with $P^{\mathcal{C}}(\xi)$ any monic polynomial of degree m. The zeros ξ_i of this polynomial are known as Chebyshev nodes.

Note that after choosing $\omega_i = 1/\xi_i$, we have that the nodes of $T_m^{\mathcal{C}}(\xi)$ and $\Omega_m(\xi) = \prod_{i=1}^m (1 - \omega_i \xi)$ coincide and hence that $\Omega_m(\mathbf{A})$ is small on $\sigma(\mathbf{A})$.

For some sets, the Chebyshev polynomial is explicitly known. If for instance the set equals $\mathcal{C} = [-1, 1]$, the Chebyshev polynomial of degree m is given by

$$T_m^{[-1,1]}(\xi) = \prod_{i=1}^{m} \left(\cos\left(\frac{2i-1}{2m} \pi \right) - \xi \right),$$

and this can easily be extended to an arbitrary interval $\mathcal{C} = [f_1, f_2]$ by a proper transformation, which results in the Chebyshev nodes

$$\xi_i = \frac{1}{2}(f_1 + f_2) + \frac{1}{2}(f_2 - f_1) \cos\left(\frac{2i-1}{2m} \pi \right). \tag{6.12}$$

For many problems, however, the Ritz values and eigenvalues do not lie on a straight line and a two-dimensional set is needed. The easiest convex set on which the Chebyshev polynomial is explicitly known is a disk $D(c, r)$. The m-th Chebyshev polynomial on this set is given by

$$T_m^{D(c,r)}(\xi) = (\xi - c)^m,$$

with Chebyshev nodes $\xi_i = c$.

The Chebyshev polynomial is also known on the area enclosed by an ellipse with foci $f_{1,2}$ and major radius a. This set can be defined by $\mathcal{C} = \{\xi \in \mathbb{C} \ : \ |\xi + f_1| + |\xi + f_2| \leq 2a\}$. The nodes of the m-th Chebyshev polynomial on this set are the same as the nodes on the line segment that connects the two foci of this ellipse, and we therefore have that the nodes are given by (6.12). We refer to [24] for the proofs on the Chebyshev polynomials and corresponding nodes on the described sets.

It is far from trivial to obtain the minimal-area enclosing ellipse for a given set of points. One of the options is to use the so-called Khachiyan algorithm, which is a linear optimization problem that finds a solution to a certain tolerance in a finite number of steps, see [28].

Chapter 7

Numerical experiments on the car problem

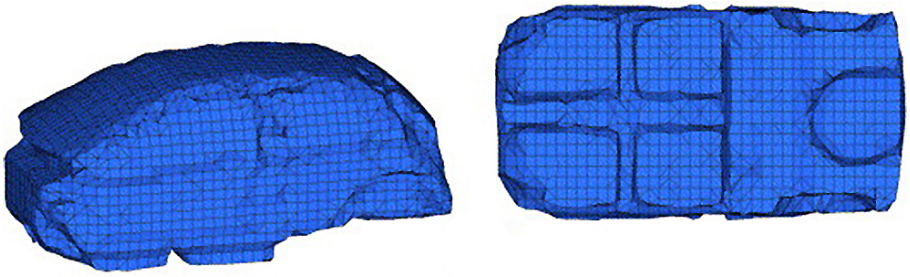
There are now three types of scientists: experimental, theoretical, and computational.
Silvan Samuel Schweber

7.1 Modelling the acoustics of a car

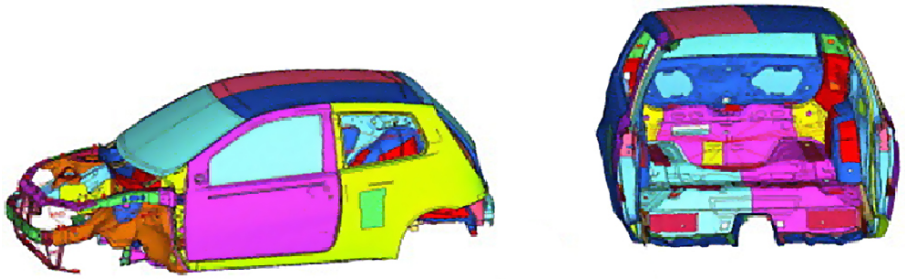
We focus on the modelling of the pressure disturbances caused by acoustic waves inside a moving car, where we refer to [14] for a detailed description. The acoustics waves arise from not only the engine, but also from vibrations of the car structure due to for instance road contact and wind. To limit the amount of noise in a car it is needed that the pressure disturbances in the air inside the car (the car fluid) and the vibrations of the car (the car structure) are investigated. We combine these vibrations in fluid and structure in a single numerical system and define to this end the car fluid as $\Omega \subset \mathbb{R}^3$, which is surrounded by the structure surface Γ and the car structure $\widetilde{\Omega}$. As an example, we consider a FIAT Punto, which is taken from [3]. In Figure 7.1 the physical car is displayed, where the corresponding fluid part Ω and structure part $\widetilde{\Omega}$ are given in Figure 7.2.

Figure 7.1: The FIAT Punto.

The boundary conditions on Γ are obtained from the interaction between the fluid and the structure of the car. The displacement of the structure is described by so-called far field conditions.



(a) fluid part



(b) structure part

Figure 7.2: The FEM model of the FIAT Punto.

The wave equation (2.6) describes the pressure disturbances of the fluid Ω and the weak formulation is given by

$$\int_\Omega \frac{1}{\rho_0}\nabla p\cdot\nabla\eta\ d\Omega+\int_\Gamma \frac{r}{(\rho_0 c_0)^2}\eta\frac{\partial p}{\partial t}\ d\Gamma+\int_\Omega \frac{1}{\rho_0 c_0^2}\eta\frac{\partial^2 p}{\partial t^2}\ d\Omega+\int_\Gamma \mathbf{n}\cdot\frac{\partial^2\mathbf{u}}{\partial t^2}\eta\ d\Gamma=0, \tag{7.1}$$

where $\eta\in H^1(\Omega)$ such that it satisfies the boundary conditions on Γ, and $\mathbf{u}=\mathbf{u}(\mathbf{x},t)$ the displacement of the structure surface Γ. Damping and absorption at the boundary is incorporated in the extra term

$$\int_\Gamma \frac{r}{(\rho_0 c_0)^2}\eta\frac{\partial p}{\partial t}\ d\Gamma,$$

and $r=r(\boldsymbol{\alpha})$ depends on the properties of the material described by $\boldsymbol{\alpha}$.

Finite element discretisation of (7.1) leads to the system

$$\mathbf{M}_\mathrm{f}\frac{d^2}{dt^2}\mathbf{p}+\mathbf{D}_\mathrm{f}\frac{d}{dt}\mathbf{p}+\mathbf{K}_\mathrm{f}\mathbf{p}+\mathbf{D}_\mathrm{sf}\frac{d^2}{dt^2}\mathbf{u}=\mathbf{0}.$$

The vibrations of the structure are obtained by applying a direct discrete finite element method [14] and we obtain the system

$$\mathbf{M}_\mathrm{s}\frac{d^2}{dt^2}\mathbf{u}+\mathbf{D}_\mathrm{s}\frac{d}{dt}\mathbf{u}+\mathbf{K}_\mathrm{s}\mathbf{u}-\mathbf{D}_\mathrm{sf}^T\mathbf{p}=\mathbf{f}_\mathrm{s}.$$

We combine these systems into

$$\begin{pmatrix}\mathbf{M}_\mathrm{s} & \mathbf{0}\\ \mathbf{D}_\mathrm{sf} & \mathbf{M}_\mathrm{f}\end{pmatrix}\frac{d^2}{dt^2}\begin{pmatrix}\mathbf{u}\\ \mathbf{p}\end{pmatrix}+\begin{pmatrix}\mathbf{D}_\mathrm{s} & \mathbf{0}\\ \mathbf{0} & \mathbf{D}_\mathrm{f}\end{pmatrix}\frac{d}{dt}\begin{pmatrix}\mathbf{u}\\ \mathbf{p}\end{pmatrix}+\begin{pmatrix}\mathbf{K}_\mathrm{s} & -\mathbf{D}_\mathrm{sf}^T\\ \mathbf{0} & \mathbf{K}_\mathrm{f}\end{pmatrix}\begin{pmatrix}\mathbf{u}\\ \mathbf{p}\end{pmatrix}=\begin{pmatrix}\mathbf{f}_\mathrm{s}\\ \mathbf{0}\end{pmatrix}. \tag{7.2}$$

We note that the matrices in (7.2) are highly dependent on the geometry of the car and the type of finite elements that are used.

We perform the Fourier Ansatz

$$\begin{pmatrix}\mathbf{u}\\ \mathbf{p}\end{pmatrix} = \begin{pmatrix}\tilde{\mathbf{u}}\\ \tilde{\mathbf{p}}\end{pmatrix} e^{i\omega t}, \mathbf{f}_{\mathrm{s}} = \tilde{\mathbf{f}} e^{i\omega t},$$

and obtain after multiplication of the second block row of (7.2) by ω^{-1} the frequency dependent linear system

$$\left\{-\omega^2 \begin{pmatrix}\mathbf{M}_{\mathrm{s}} & \mathbf{0}\\ \mathbf{0} & \mathbf{M}_{\mathrm{f}}\end{pmatrix} + i\omega \begin{pmatrix}\mathbf{D}_{\mathrm{s}} & i\mathbf{D}_{\mathrm{sf}}^T\\ i\mathbf{D}_{\mathrm{sf}} & \mathbf{D}_{\mathrm{f}}\end{pmatrix} + \begin{pmatrix}\mathbf{K}_{\mathrm{s}} & \mathbf{0}\\ \mathbf{0} & \mathbf{K}_{\mathrm{f}}\end{pmatrix}\right\} \begin{pmatrix}\tilde{\mathbf{u}}\\ \tilde{\mathbf{p}}/\omega\end{pmatrix} = \begin{pmatrix}\tilde{\mathbf{f}}\\ \mathbf{0}\end{pmatrix},$$

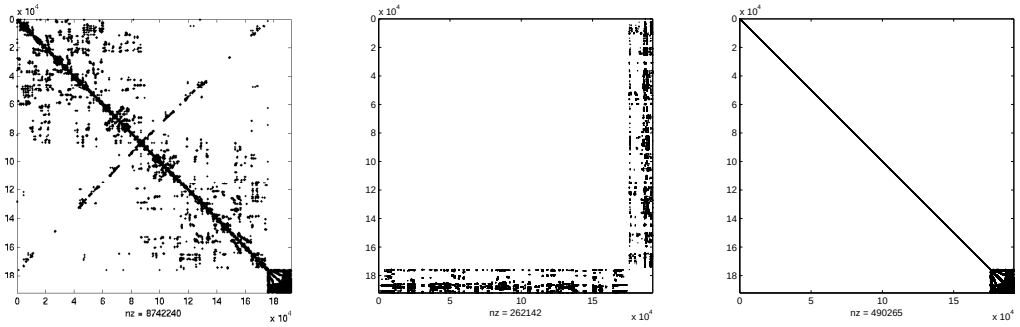
or, for short

$$\{\mathbf{K} + f \cdot 2i\pi\mathbf{D} - f^2 \cdot 4\pi^2\mathbf{M}\}\mathbf{x} = \mathbf{A}(f)\mathbf{x} = \mathbf{b}. \tag{7.3}$$

The matrices $\mathbf{K}$, $\mathbf{D}$ and $\mathbf{M}$ are symmetric and sparse. The goal is to solve the system (7.3) for the range of frequencies $f = 1, 2, \ldots$ Hz.

7.2 The mathematical problem

We consider two numerical versions of the linear system (7.3), which are constructed by the engineering company SFE in Berlin. The first consists of $n = 192\,184$ and the second of $n = 495\,151$ equations, where the fluid part has $n_{\mathrm{f}} = 16\,363$ unknowns and the structure part either $n_{\mathrm{s}} = 175\,821$ or $n_{\mathrm{s}} = 478\,788$ unknowns. We will focus on the smaller of these two. Unfortunately, we do not know what the individual entries stand for and we cannot visualise any solution vector $\mathbf{x}$. The matrix structures of the smaller problem are given in Figure 7.3. The matrix structures for the larger one are very similar to these structures.



(a) Stiffness matrix $\mathbf{K}$. (b) Damping matrix $\mathbf{D}$. (c) Mass matrix $\mathbf{M}$.

Figure 7.3: Matrix structures.

We first focus on IDR(s), after which we compare the resulting best strategy for this method with other well-known Krylov subspace methods. All experiments are performed on a desktop PC with a single CPU clocked at 3.0 GHz and with 16 Gb RAM with Matlab 7.7, where we used the standard Matlab implementations for GMRES, CGS and BiCGSTAB and for IDR(s) the Matlab code of M.B. van Gijzen[1]. We consider several (modified) shifted Laplace preconditioners and

[1]IDR(s) has been included in the Collected Algorithms of the ACM as Algorithm 913, see http://calgo.acm.org

different approaches in reusing the available information that results from the computations for other frequencies that are carried out earlier.

As we have seen, the car problem consists of a fluid and a structure part with interaction terms in the damping matrix $\mathbf{D}$. First, we focus on the fluid part and structure part separately and then we consider the complete problem. The fluid part of the car problem, in short the fluid problem, consists of real numbers only and is given by

$$(\mathbf{K}_\mathrm{f} - f^2 \cdot 4\pi^2 \mathbf{M}_\mathrm{f})\mathbf{x}_\mathrm{f} = \mathbf{A}_\mathrm{f}(f)\mathbf{x}_\mathrm{f} = \mathbf{b}_\mathrm{f}, \tag{7.4}$$

where we fix the right hand side $\mathbf{b}_\mathrm{f} = \mathbf{e}_{\lceil 0.5 n_\mathrm{f} \rceil}$. That is, we consider a point source in one of the nodes. The damping at the boundary is incorporated in the interaction term $-\omega \mathbf{D}_\mathrm{sf}$ and hence $\mathbf{D}_\mathrm{f} = \mathbf{0}$. The structure problem equals

$$(\mathbf{K}_\mathrm{s} + f \cdot 2i\pi \mathbf{D}_\mathrm{s} - f^2 \cdot 4\pi^2 \mathbf{M}_\mathrm{s})\mathbf{x}_\mathrm{s} = \mathbf{A}_\mathrm{s}(f)\mathbf{x}_\mathrm{s} = \mathbf{b}_\mathrm{s}. \tag{7.5}$$

The stiffness matrix $\mathbf{K}_\mathrm{s}$ and the damping matrix $i\omega \mathbf{D}_\mathrm{s}$ are complex matrices. This last matrix contains only 82 non-zeros. As right hand side we consider again a point source in a node and we choose $\mathbf{b}_\mathrm{s} = \mathbf{e}_{\lceil 0.5 n_\mathrm{s} \rceil}$. It follows from Figure 7.3(c) that $\mathbf{M}_\mathrm{s}$, the mass matrix of the structure problem, is almost diagonal. In fact, the upper and lower bandwidth equal 5 and hence $\mathbf{M}_\mathrm{s}$ consists only of 11 non-zero diagonals.

For all experiments we set tol $= 10^{-8}$ and maxit $= 1000$, unless it is explicitly stated otherwise.

7.3 LU factorisations of the preconditioners

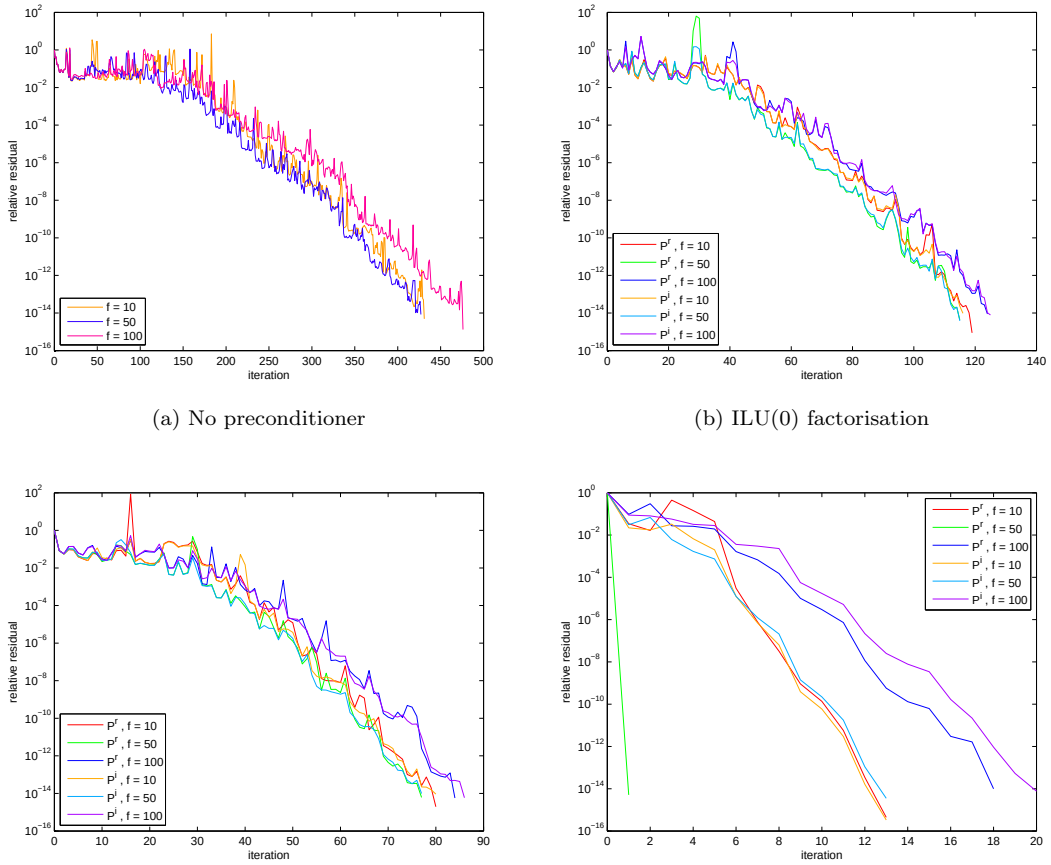
In the first set of experiments, we use the shifted Laplace preconditioner to solve the fluid part (7.4) and structure part (7.5) of the car problem. We consider the preconditioner with a real shift and an imaginary shift with a fixed shift frequency f_0 and apply it to problems with several frequencies. The shifted Laplace preconditioner with a purely imaginary shift is defined as

$$\mathbf{P}^i(f_0) = \mathbf{K} + f_0 \cdot 2i\pi \mathbf{D} + f_0^2 \cdot 4i\pi^2 \mathbf{M}, \tag{7.6}$$

and the shifted Laplace preconditioner with a real shift is given by

$$\mathbf{P}^r(f_0) = \mathbf{K} + f_0 \cdot 2i\pi \mathbf{D} - f_0^2 \cdot 4\pi^2 \mathbf{M} = \mathbf{A}(f_0). \tag{7.7}$$

The reduction to similar preconditioners for the separate fluid and structure problem follows naturally and is indicated by a subscripted 'f' for the fluid and 's' for the structure problem. We factorise the preconditioners with three types of (incomplete) LU factorisation, namely exact LU factorisation, ILU(0) factorisation and ILUT(τ) factorisation with $\tau = 0.01$. We investigate if the preconditioners and the factorisations for the fluid and the structure problem lead to results that are comparable to the results that we obtained earlier for the room problem, see Appendix B. We use IDR(4), choose as initial guess $\mathbf{x}_0 = \mathbf{0}$ and consider the relative residuals $\|\mathbf{r}_i\|/\|\mathbf{b}\|$ at the i-th iteration for the frequencies $f = 10$, 50 and 100 Hz. In Figure 7.4 the results for the fluid problem are displayed. We compare the results for the unpreconditioned problem, given in Figure 7.4(a), to the problem where we apply the factorised preconditioners $\mathbf{P}_\mathrm{f}^r(f_0)$ and $\mathbf{P}_\mathrm{f}^i(f_0)$ with $f_0 = 50$ Hz with different (incomplete) LU factorisations, see Figure 7.4(b)-(d). Some results concerning computation time and number of iterations to obtain a relative accuracy of the residual that satisfies $\|\mathbf{r}_i\|/\|\mathbf{b}\| \leq$ tol are tabulated in Table 7.1.



(a) No preconditioner

(b) ILU(0) factorisation

(c) ILUT(0.01) factorisation

(d) LU factorisation

Figure 7.4: Relative residual norms per iteration for the fluid problem.

		Computation time (s)				Number of iterations		
Precond.	Factorisation	Factor.	10 Hz	50 Hz	100 Hz	10 Hz	50 Hz	100 Hz
$\mathbf{P}_{\mathrm{f}}^{r}(50)$	LU	1 351.23	79.88	17.65	121.71	9	1	13
	ILUT(0.01)	5.26	1.45	1.40	1.49	58	56	65
	ILU(0)	0.07	0.66	0.62	0.71	86	84	95
$\mathbf{P}_{\mathrm{f}}^{i}(50)$	LU	3 282.47	69.96	70.07	108.95	9	9	14
	ILUT(0.01)	10.06	2.13	1.90	2.25	59	55	63
	ILU(0)	0.11	1.15	1.04	1.22	86	84	94
No preconditioning		-	1.26	1.20	1.31	321	311	342

Table 7.1: Computational results at a residual norm of 10^{-8} for the fluid problem.

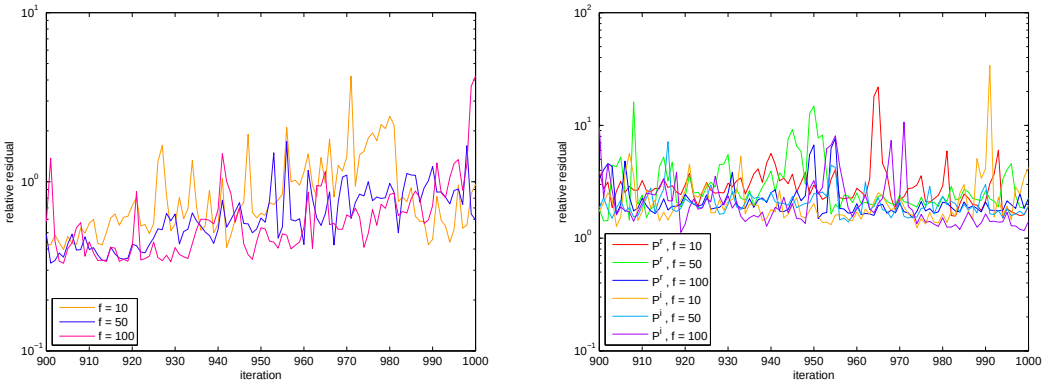
We have seen that for the room problem the number of iterations that is needed to obtain a certain accuracy is independent from the shift of the preconditioner f_0, as soon as we use incomplete LU

factorisation to factor a preconditioner. The results from Figure 7.4(b)-(c) strongly suggest that this also holds for the fluid problem, since the residuals of the problems with system matrices $[\mathbf{P}_{\mathrm{f}}^{r}(f_0)]^{-1}\mathbf{A}_{\mathrm{f}}(f)$ and $[\mathbf{P}_{\mathrm{f}}^{i}(f_0)]^{-1}\mathbf{A}_{\mathrm{f}}(f)$ almost coincide for the same frequency f while the shifts of these two preconditioners are in fact completely different from each other. We also see in Table 7.1 that the numbers of iterations for a relative residual norm of 10^{-8} are identical or at least very similar for both preconditioners. If we use $\mathbf{P}_{\mathrm{f}}^{i}(f_0)$ as preconditioner, the factors contain complex numbers and the computation time is greater in comparison with the use of $\mathbf{P}_{\mathrm{f}}^{r}(f_0)$, which can be explained by the fact that the fluid problem contains real numbers only.

The exact LU factorisation results in the need of very few iterations for a proper accuracy, but computing and applying the preconditioners is very expensive. More generally, we see that the application of any preconditioner reduces the number of iterations greatly and that the better the product of the factors resembles the preconditioner, the less iterations are needed. However, the application of a preconditioner increases the computation time per iteration drastically. We see confirmed that the higher the number of non-zeros in the (incomplete) LU factors, the larger the computation time. In this context, we note that solving the problem without preconditioning needs by far the most iterations, but the computation time that is needed is surprisingly small compared to the cases where we use a preconditioner.

We consider the same approach for the structure problem. However, it is only possible to compute the ILU(0) factors of the preconditioners in a reasonable amount of time and with relatively small storage requirements. We use IDR(4), fix $\mathbf{x}_0 = \mathbf{0}$ for the frequencies $f = 10, 50, 100$ Hz and as frequency for the shifted Laplace preconditioners we choose again $f_0 = 50$ Hz.

Whether we apply no preconditioner or we use the ILU(0) factors, we do not observe convergence for any frequency. In Figure 7.5 the residual norm is given after 900 to 1000 iterations and we see that it behaves erratic for all frequencies.



(a) No preconditioning

(b) ILU(0) factorisation

Figure 7.5: Relative residual norms per iteration for the structure problem.

7.3.1 Pivoting

While the computation time for the exact LU factors for the fluid problem is already very high, it is impossible to construct the factors for the structure problem on an average desktop computer. This is caused by the fact that the standard Matlab routine keeps interchanging rows during the factorisation to improve the accuracy of the resulting factors. This interchanging is based on a certain element in this matrix, the pivot element, which is interchanged with a more desirable entry. So instead of determining the LU factors such that $\mathbf{A} = \mathbf{LU}$, the Matlab routine produces in fact a permuted lower triangular matrix $\mathbf{P}^{-1}\mathbf{L}$ and an upper triangular matrix $\mathbf{U}$. The matrix $\mathbf{P}$ is a permutation matrix[2], and hence it can be written as $(\mathbf{e}_{k_1}^T, \mathbf{e}_{k_2}^T, \ldots, \mathbf{e}_{k_n}^T)^T$ with $(k_1, k_2 \ldots, k_n)$ a permutation of $(1, 2, \ldots, n)$. Note that the product $\mathbf{PA}$ consists of the rows of $\mathbf{A}$ rearranged in a new order and, similarly, the product $\mathbf{AP}$ is equal to a reordering of the columns of $\mathbf{A}$.

For our problems, the computation time increases so immensely, that we cannot use the standard Matlab routine. Therefore, we suppress the standard way of pivoting and control the interchanging of rows and columns in the process of factorisation by fixing two thresholds, τ_1 and τ_2, in the determination of LU factors that satisfy $\mathbf{PAQ} = \mathbf{LU}$, with $\mathbf{P}$ and $\mathbf{Q}$ permutation matrices. The thresholds are for symmetric matrices as follows: a diagonal element is chosen such that $|\mathbf{A}_{j,j}| \geq \tau_1 \max |\mathbf{A}_{j:n,j}|$. If no diagonal element satisfies this requirement, an element on the subdiagonal is selected such that $|\mathbf{A}_{j+1,j}| \geq \tau_2 \max |\mathbf{A}_{j+1:n,j}|$. The standard choice of thresholds are $\tau_1 = 10^{-3}$ and $\tau_2 = 0.1$.

If we set $\tau_1 = 0$ (and $\tau_2 = 0$), we perform the LU factorisation with so-called diagonal pivoting: we choose as pivot element the largest remaining diagonal element, that is, $|\mathbf{A}_{j,j}| \geq \max_{i \geq j} |\mathbf{A}_{i,i}|$. This means that $\mathbf{P} = \mathbf{Q}^T$ and for symmetric matrices also that $\mathbf{L}$ and $\mathbf{U}^T$ have the same sparsity pattern.

			Computation time (s)	
Problem	Preconditioner	Thresholds	Factorisation	Application
Fluid	$\mathbf{P}_{\mathrm{f}}^r(f_0)$	None	1 355.71	8.93
		$\tau_1 = 10^{-3}, \tau_2 = 0.1$	1.05	0.03
		$\tau_1 = 0, \tau_2 = 0$	1.04	0.03
	$\mathbf{P}_{\mathrm{f}}^i(f_0)$	None	3 322.32	8.30
		$\tau_1 = 10^{-3}, \tau_2 = 0.1$	2.54	0.08
		$\tau_1 = 0, \tau_2 = 0$	2.55	0.08
Structure	$\mathbf{P}_{\mathrm{s}}^r(f_0)$	$\tau_1 = 10^{-3}, \tau_2 = 0.1$	50.13	0.94
		$\tau_1 = 0, \tau_2 = 0$	36.24	0.80
	$\mathbf{P}_{\mathrm{s}}^i(f_0)$	$\tau_1 = 10^{-3}, \tau_2 = 0.1$	49.42	0.92
		$\tau_1 = 0, \tau_2 = 0$	36.60	0.80

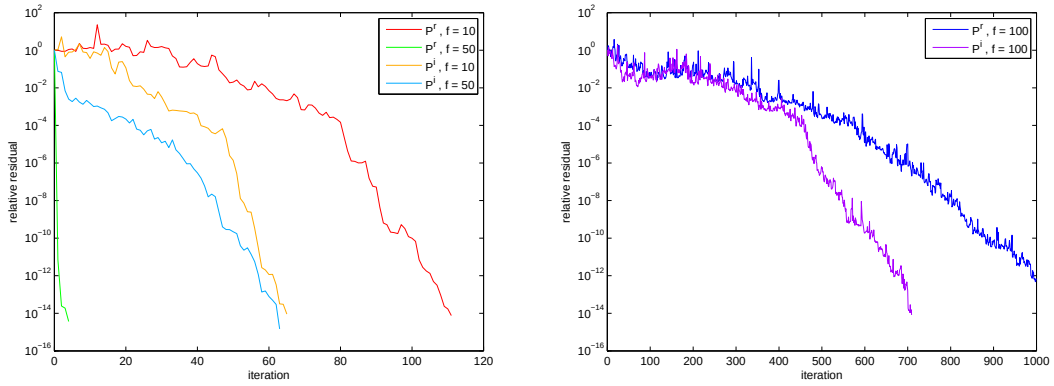
Table 7.2: Exact LU factorisation for the fluid and structure problem.

If we use either of the above thresholds, the computation and application time reduces spectacularly for the fluid problem and we are also able to determine exact LU factors for the preconditioners of the structure problem. For a comparison of the LU factorisations, see Table 7.2. There is hardly any difference in the choice of thresholds for the fluid problem, but for the structure

[2]Note that in this context $\mathbf{P}$ is not a preconditioner.

problem both the factorisation and application time is less for diagonal pivoting and hence we choose this type of pivoting in all upcoming experiments.

We can now solve the earlier described structure problem, since the residual norms do converge when exact LU factorisation with pivoting is used. We repeat the above experiment where we now use LU factorisation with diagonal pivoting and the resulting relative residual norms are given in Figure 7.6. The computation time and number of iterations for an accuracy of 10^{-8} are tabulated in Table 7.3. We note that for increasing frequency the number of iterations becomes very large.



(a) LU factorisation for f = 10, 50 Hz

(b) LU factorisation for f = 100 Hz

Figure 7.6: Relative residual norms per iteration for the structure problem.

	Computation time (s)			Number of iterations		
Preconditioner	10 Hz	50 Hz	100 Hz	10 Hz	50 Hz	100 Hz
$\mathbf{P}^r(50)$	80.32	1.00	699.07	91	1	794
$\mathbf{P}^i(50)$	47.04	40.81	483.06	53	46	549

Table 7.3: Computational results at a residual norm of 10^{-8} for the fluid structure.

7.4 Solving sequences of fluid and structure problems

As we have shown in the preceding section, we are able to separately solve the fluid and structure problems, at least for small frequencies. In this section we investigate ways to reduce the computation time by the use of earlier obtained solutions and by modifying the shifted Laplace preconditioners (7.6) and (7.7).

First, we consider the reuse of information for the initial guess, by extrapolation the solutions that we obtained earlier or by spanning an initial search space $\mathcal{U}_0$. Second, we investigate the performance of the shifted Laplace preconditioners $\mathbf{P}_*^r(f_0)$ and $\mathbf{P}_*^i(f_0)$ and for the structure problem a modified shifted Laplace preconditioner $\mathbf{P}_{\mathrm{s}}^m(f_0)$ which equals $\mathbf{P}_{\mathrm{s}}^r(f_0)$ except that we ignore the complex part. Next, we investigate if it is reasonable to update the shift in the shifted Laplace preconditioners and if so, how we should choose the moment of updating and which shift the

new preconditioner should have. We also compare the number of iterations that different Krylov subspace methods need to obtain the same accuracy, similar to the experiments we did for the room problem. We consider the Krylov subspace methods that we described in sections 4.3 and 4.4 and IDR(s) for different values of s.

7.4.1 Reuse of solution vectors

Extrapolation of previous solution vectors for the initial guess $\mathbf{x}_0$

One of the ways to reduce the number of iterations that a numerical method needs for convergence to a certain tolerance, is to improve the initial guess for the problem. The standard choice is $\mathbf{x}_0 = \mathbf{0}$, but some (educated) guess could be much better. Since we solve sequences of problems, the solutions to previously solved problems are available. It is reasonable to use some of these previous solutions for a maybe better initial guess, by using some type of extrapolation. We investigate if cubic spline and Lagrange extrapolation of at most s solutions reduce the computation time for the fluid problem for both IDR(4) and IDR(8) with $\mathbf{P}_{\mathrm{f}}^{i}(f_0)$, where $f_0 = 50$ Hz, for the frequencies $1, 2, \ldots, 100$ Hz. We note that a similar experiment with $\mathbf{P}_{\mathrm{f}}^{r}(f_0)$ as preconditioner leads to practically the same reduction in numbers of iterations for higher frequencies in the fluid problem and for the structure problem. For now, we do not use an initial search space and we emphasize that we use for IDR(4) the previous 4 solutions and for IDR(8) the previous 8 solutions in order to make a fair comparison with the choices where we do use the initial search space that contains s vectors. We compare the results for cubic spline and Lagrange extrapolation with choices of the starting vector $\mathbf{x}_0$ equal to $\mathbf{0}$ (no extrapolation), equal to the last solution (constant extrapolation) and based on linear extrapolation of the previous two solutions. The results are given in Figure 7.7(a)-(b). Figure 7.7(c)-(d) result from the same experiment with a shift for the preconditioners $f_0 = 525$ Hz and a frequency range $501, 502, \ldots, 550$ Hz. The total number of iterations is given in Table 7.4. In order to keep a good overview, we summerise the choices for the experiment.

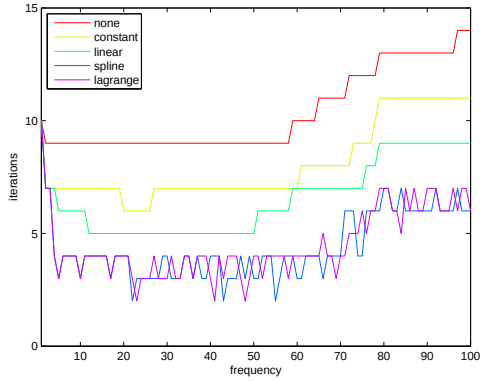
- Objective: performance of several types of extrapolation.
- Problem: the fluid problem.
- Frequency range: $f = 1, 2, \ldots 100$ Hz and $f = 501, 502, \ldots 550$ Hz.
- Method: IDR(s) with $s = 4, 8$.
- Preconditioner: $\mathbf{P}_{\mathrm{f}}^{i}(f_0)$ with $f_0 = 50$ Hz and $f_0 = 525$ Hz.
- Results: Figure 7.7 and Table 7.4.

		Total number of iterations				
Frequency range	Method	None	Constant	Linear	Cubic spline	Lagrange
$f = 1, 2, \ldots, 100$ Hz.	IDR(4)	1 034	809	657	442	451
	IDR(8)	997	740	617	449	593
$f = 501, 502, \ldots, 550$ Hz.	IDR(4)	25 316	24 249	23 263	21 718	21 729
	IDR(8)	18 745	18 172	17 442	16 232	16 274

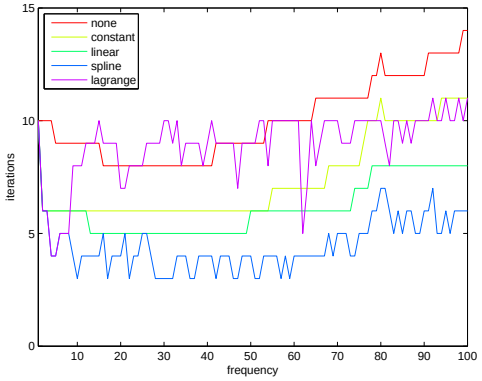
Table 7.4: Number of iterations for the fluid problem with different types of extrapolation.

The improvements are significant, although for the frequency range $f = 501, 502, \ldots, 550$ Hz the relative improvement is much less for all types of extrapolation compared to the reduction on the range $f = 1, 2 \ldots, 100$ Hz. The higher the order of the extrapolation, the better the number of iterations is reduced and cubic spline and Lagrange extrapolation reduce the number of iterations

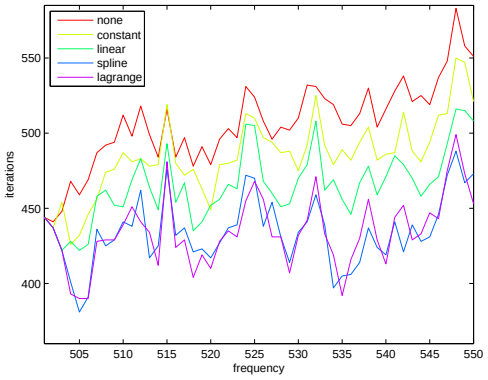
the most, except for IDR(8) for the frequencies $f = 1, 2, \ldots, 100$ Hz. The difference between these two in all other test cases is per frequency erratic and is over a larger range of frequencies negligible.



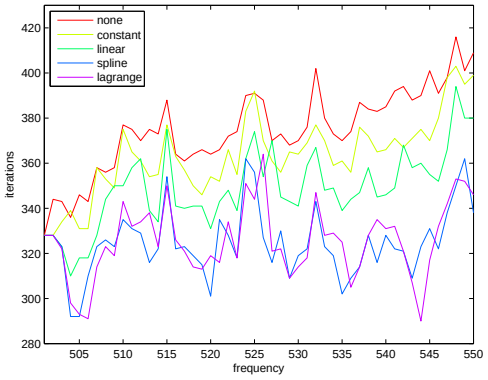
(a) IDR(4), $f = 1, 2, \ldots, 100$ Hz.



(b) IDR(8), $f = 1, 2, \ldots, 100$ Hz.



(c) IDR(4), $f = 501, 502, \ldots, 550$ Hz.



(d) IDR(8), $f = 501, 502, \ldots, 550$ Hz.

Figure 7.7: Number of iterations for the fluid problem with different types of extrapolation.

If we compare the results for $f = 1, 2, \ldots, 100$ Hz with the results of $f = 501, 502, \ldots, 550$ Hz, we see that the number of iterations that are needed for convergence becomes very large for higher frequencies. If we apply IDR(4) with Lagrange interpolation, we need on average 5 iterations per frequency for $f = 1, 2 \ldots, 100$ Hz and more than 430 iterations for $f = 501, 502, \ldots, 550$ Hz. For IDR(8) and Lagrange interpolation this is subsequently 6 and 325 iterations. This means that for small frequencies IDR(4) is slightly better than IDR(8), but IDR(8) clearly outperforms IDR(4) for higher frequencies and hence more difficult problems.

We repeat the same experiment for the structure problem, but restrict ourself to the sequence of $f = 1, 2 \ldots, 100$ Hz since for frequencies above 500 Hz, we do not observe convergence within 1000 iterations. The number of iterations per frequency is displayed in Figure 7.8, while the total number of iterations is given in Table 7.5. In short:

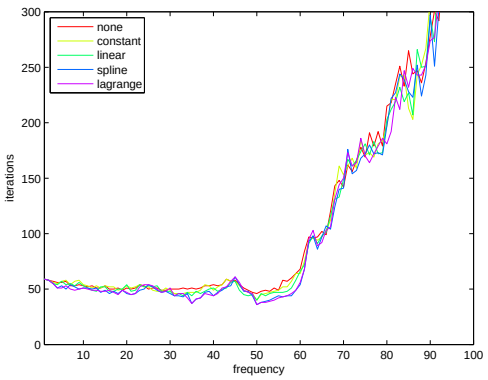
- Objective: performance of several types of extrapolation.

- Problem: the structure problem.
- Frequency range: $f = 1, 2, \ldots 100$ Hz.
- Method: IDR(s) with $s = 4$ and $s = 8$.
- Preconditioner: $\mathbf{P}_{\mathrm{s}}^{i}(f_0)$ with $f_0 = 50$ Hz.
- Results: Figure 7.8 and Table 7.5.

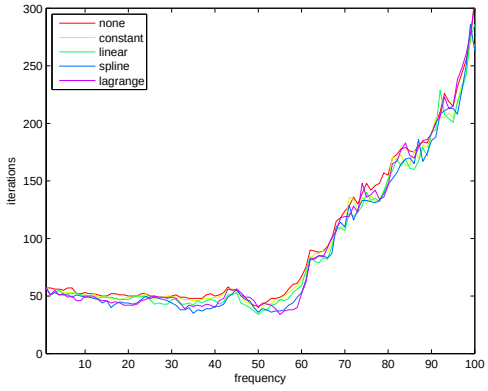
		Total number of iterations				
Frequency range	Method	None	Constant	Linear	Cubic spline	Lagrange
$f = 1, 2, \ldots, 100$ Hz.	IDR(4)	12 654	12 395	12 208	11 953	11 931
	IDR(8)	9 667	9 249	9 030	9 205	8 867

Table 7.5: Number of iterations for the structure problem with different types of extrapolation.

It follows that for higher order extrapolation the reduction in the number of iterations is larger, but the differences are relatively small. For small frequencies, IDR(8) and IDR(4) are again more or less comparable and the number of iterations is small, while IDR(8) performs much better for frequencies larger than 50 Hz, where the number of iterations increases significantly. IDR(4) needs up to 600 iterations for the frequencies close to 100 Hz.



(a) IDR(4), $f = 1, 2, \ldots, 100$ Hz.



(b) IDR(8), $f = 1, 2, \ldots, 100$ Hz.

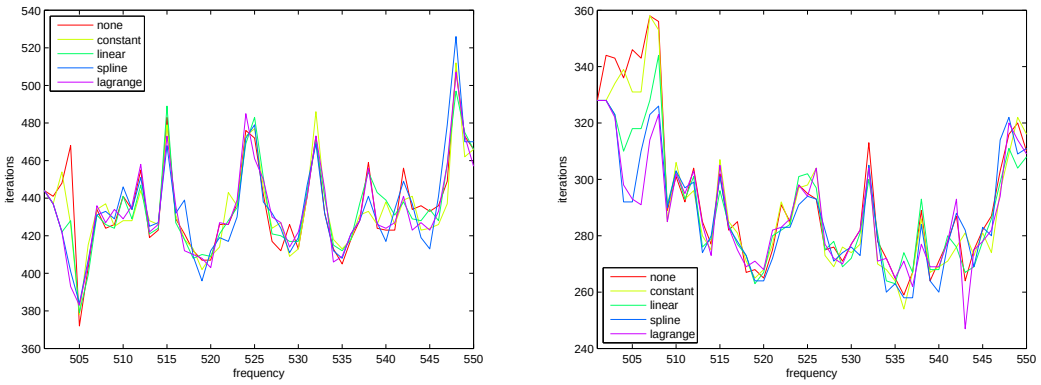
Figure 7.8: Number of iterations for the structure problem with different types of extrapolation.

The initial search space $\mathcal{U}_0$

We investigate if an initial search space $\mathcal{U}_0$ reduces the number of iterations. We therefore repeat the above experiments for the fluid and structure problem, but now we define for the problem with frequency f the space $\mathcal{U}_0$ as the span of the last s solutions $\mathbf{x}^{f-1}, \mathbf{x}^{f-2}, \ldots, \mathbf{x}^{f-s}$. Some results for the fluid problem are presented per frequency in Figure 7.9 and the totals are tabulated in Table 7.6. We give a short summary of this experiment.

- Objective: performance of $\mathcal{U}_0$.
- Problem: the fluid problem.
- Frequency range: $f = 1, 2, \ldots 100$ Hz and $f = 501, 502, \ldots 550$ Hz.

- Method: IDR(s) with $s = 4$ and $s = 8$.
- Preconditioner: $\mathbf{P}_{\mathrm{f}}^{i}(f_0)$ with $f_0 = 50$ Hz and $f_0 = 525$ Hz.
- Results: Figure 7.9 and Table 7.6.



(a) IDR(4), $f = 501, 502, \ldots, 550$ Hz.

(b) IDR(8), $f = 501, 502, \ldots, 550$ Hz.

Figure 7.9: Number of iterations for the fluid problem with initial search space $\mathcal{U}_0$.

		Total number of iterations per type of extrapolation				
Frequency range	Method	None	Constant	Linear	Cubic spline	Lagrange
$f = 1, \ldots, 100$ Hz	IDR(4)	657	648	646	645	645
	IDR(8)	884	894	862	868	895
$f = 501, \ldots, 550$ Hz	IDR(4)	21 768	21 718	21 732	21 708	21 673
	IDR(8)	14 689	14 584	14 477	14 386	14 385

Table 7.6: Number of iterations for the fluid problem with initial search space $\mathcal{U}_0$.

If we compare the results of this experiment with the experiment where we use extrapolation and no initial search space, see Table 7.4, we see that cubic spline and Lagrange extrapolation without the use of $\mathcal{U}_0$ have the best performance on $f = 1, 2, \ldots, 100$ Hz. For higher frequencies and hence more difficult problems, we see that applying $\mathcal{U}_0$ does reduce the number of iterations. For IDR(4) and cubic spline or Lagrange extrapolation, this improvement is negligible, but for other choices it is significant.

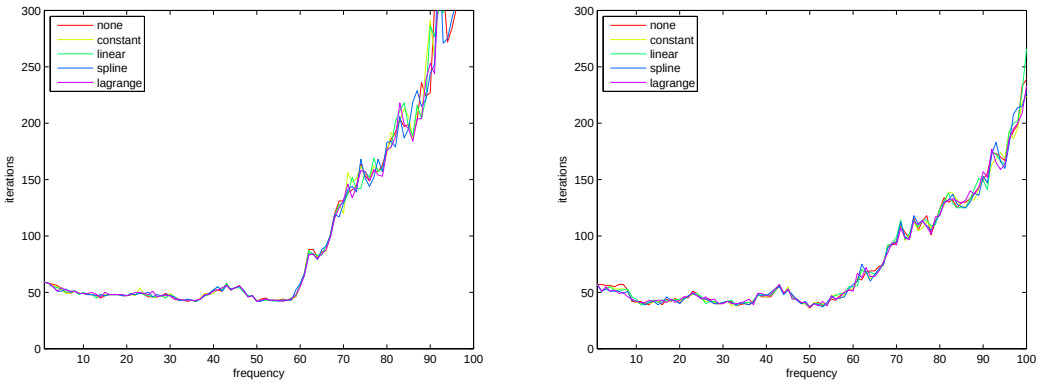
The resulting number of iterations almost coincide for both the ranges $f = 1, 2, \ldots, 100$ Hz and $f = 501, 502, \ldots, 550$ Hz. This implies that the type of extrapolation is in fact not relevant and that the extra information we use for the extrapolation is somehow already incorporated in the initial search space. The small differences in the totals are mainly caused by the differences in number of iterations for the first s frequencies. We also note that IDR(8) performs worse compared to IDR(4) for the first 100 frequencies, but that IDR(8) is much better for the much more difficult problems with frequency range $f = 501, 502, \ldots, 550$ Hz.

The results for the structure problem for $f = 1, 2, \ldots, 100$ Hz are given in Figure 7.10 and Table 7.7. We summerise this experiment as

- Objective: performance of $\mathcal{U}_0$.
- Problem: the structure problem.
- Frequency range: $f = 1, 2, \ldots 100$ Hz.
- Method: IDR(s) with $s = 4$ and $s = 8$.
- Preconditioner: $\mathbf{P}_s^i(f_0)$ with $f_0 = 50$ Hz.
- Results: Figure 7.10 and Table 7.7.

		Total number of iterations per type of extrapolation				
Frequency range	Method	None	Constant	Linear	Cubic spline	Lagrange
$f = 1, \ldots, 100$ Hz	IDR(4)	11 045	11 159	11 180	10 969	10 997
	IDR(8)	7 806	7 739	7 838	7 753	7 745

Table 7.7: Number of iterations for the structure problem with initial search space $\mathcal{U}_0$.



(a) IDR(4), $f = 1, \ldots, 100$ Hz

(b) IDR(8), $f = 1, \ldots, 100$ Hz

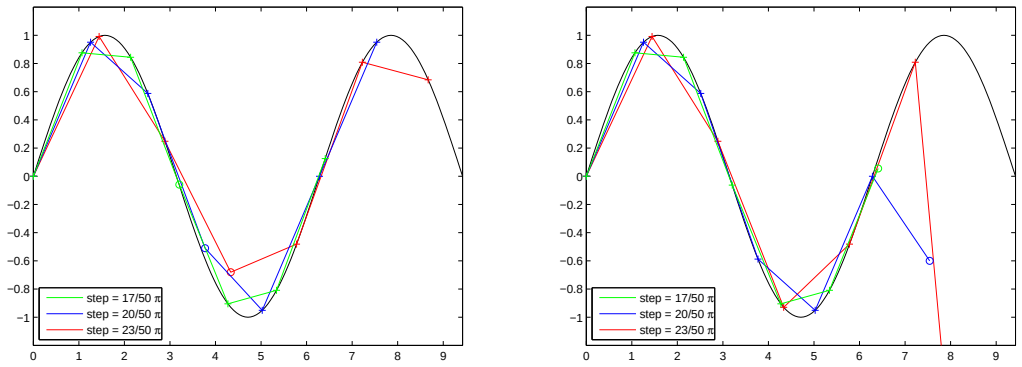
Figure 7.10: Number of iterations for the structure problem with initial search space $\mathcal{U}_0$.

The application of the initial search space $\mathcal{U}_0$ has also for the structure problem the effect that the type of extrapolation becomes irrelevant, since the total numbers of iterations are again practically the same. If we compare these results to the similar experiment without $\mathcal{U}_0$, see Table 7.5, we conclude that using the initial search space reduces the number of iterations for both IDR(4) and IDR(8) and hence is the better choice.

Inter- and extrapolation

We have seen that increasing the order of extrapolation implies that the resulting initial guess becomes better, since less iterations are needed before convergence occurs. However, the results of extrapolation are often subject to greater uncertainty compared to interpolation. For a motivation of this statement, we refer to Figure 7.11. Here, we use Lagrange interpolation and extrapolation with six given points (marked with a '+') to approximate a seventh point (marked with an 'o') on

a sine function for several step sizes. For a step size equal to $17/50\pi$, the extrapolation happens to be good, but when we choose a step size $23/50\pi$, the result is very inaccurate. If we compare extrapolation and interpolation with equal step sizes, we see that the resulting extrapolation points are all less accurate. Therefore, we should most likely prefer to use interpolation for the initial guess.



(a) interpolation (b) extrapolation

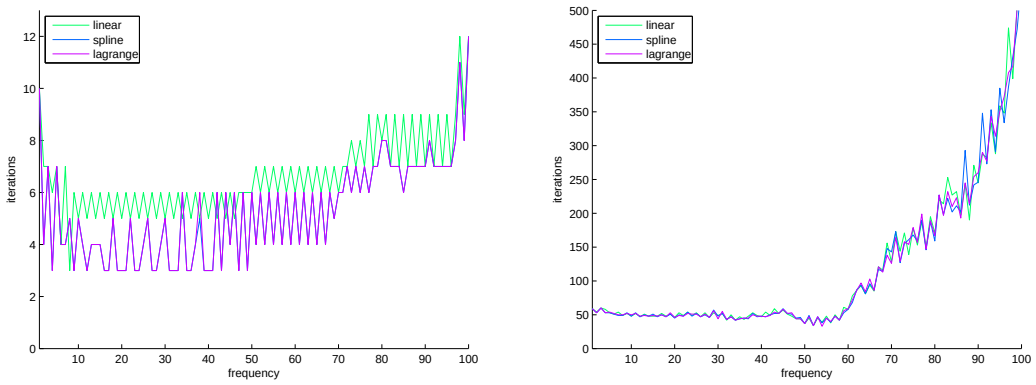
Figure 7.11: Lagrange inter- and extrapolation on a sine function.

We examine if interpolation does improve the results by combining inter- and extrapolation as follows: we use extrapolation for all odd frequencies and interpolation for the even frequencies with $f = 1, 2, \ldots, 100$ Hz. We implement this in such a way, that each set of s odd solutions is used twice: once for the extrapolation of the next odd frequency and once for the interpolation of an even frequency, where half of the known frequencies is smaller and the other half larger than this frequency. The question is: does the reduction in the number of iterations on account of the interpolation outweigh the expected increase of the number of iterations due to the extrapolation over larger step sizes?

The answer to this question is determined for the case where we use IDR(4) and as preconditioner $\mathbf{P}_*^i(f_0)$ with $f_0 = 50$ Hz and we do not use an initial search space. As a check, we consider also the case with linear inter- and extrapolation, which is of course based on 2 solutions only. The results are given in Figure 7.12 and Table 7.8.

- Objective: performance of inter- and extrapolation.
- Problem: the fluid and the structure problem.
- Frequency range: $f = 1, 2, \ldots 100$ Hz.
- Method: IDR(s) with $s = 4$.
- Preconditioner: $\mathbf{P}_*^i(f_0)$ with $f_0 = 50$ Hz.
- Results: Figure 7.12 and Table 7.8.

We first note that the results with cubic spline inter- and extrapolation are very similar to the results for the Lagrangian case, especially for the fluid problem, where the numbers of iterations almost always coincide. We also remark that the resulting curves exhibit a sawtooth-like shape. For the fluid problem sometimes the extrapolation is better and sometimes the interpolation, while for the structure problem the interpolated frequencies converge always in less iterations.



(a) fluid problem

(b) structure problem

Figure 7.12: Number of iterations with a combination of inter- and extrapolation.

	Total number of iterations		
Problem	Linear	Cubic spline	Lagrange
Fluid	661	528	529
Structure	11 827	11 662	11 931

Table 7.8: Number of iterations with a combination of inter- and extrapolation.

If we compare these results with a similar case where we only use extrapolation, see Tables 7.4 and 7.5, it follows that for the fluid problem the reduction of iterations is much better for the case with only extrapolation. For the structure problem, the results are more or less equal. This means that there is no significant improvement with the approach of inter- and extrapolation. Apparently, the extrapolation over a larger interval spoils the small improvement of interpolation to such an extent, that there is no net improvement.

Intermediate conclusions

We draw the following intermediate conclusions, based on the experiments in this section:

- The fluid problem is much easier to solve than the structure problem. For the fluid problem, the number of iterations is very small for low frequencies and we are also able to solve the problem for higher frequencies within an acceptable amount of iterations.
- For higher frequencies, both problems become much more difficult. This is caused by the fact that for increasing frequencies the system becomes more and more indefinite. The fluid problem can be solved for all frequencies smaller than 550 Hz, but we are not able to solve the structure problem within 1000 iterations for frequencies above 150 Hz with a shift f_0 for the preconditioner that is not close to f.
- Using information of previous solutions does reduce the computation time. We can use these solutions to improve the initial guess (by extrapolation of the solutions) or as a span for an initial search space $\mathcal{U}_0$. The reduction due to extrapolation is smaller than the improvement that results from the use of the initial search space for the fluid problem with high frequencies

and for the structure problem, that is, for the difficult problems. If we use both strategies, it follows that the type of extrapolation becomes irrelevant (after s iterations) as soon as we choose this space $\mathcal{U}_0$.

- IDR(8) leads to much better results compared to IDR(4) for the higher frequencies. A question that remains is, if further increasing of s will improve the computational results. This comes with higher storage requirements, since more vectors (of size n) need to be stored. However, since the LU factorisation is a much more demanding operation, we do not consider this to be a problem for $s \leq 32$. We will come back to this later.

7.4.2 A modified shifted Laplace preconditioner

Comparison of preconditioners

We have seen that the construction and application of a preconditioner that contains only real numbers is faster for the fluid problem (see Table 7.2). Therefore, we investigate for the structure part a modified real shifted Laplace preconditioner, by discarding all the 344 627 complex terms out of a total of 8 742 240 non-zeros in $\mathbf{K}_\mathrm{s}$ and the complete damping matrix $\mathbf{D}_\mathrm{s}$, which consists only of 82 non-zeros. The resulting modified shifted Laplace preconditioner consists of real numbers and is defined as

$$\mathbf{P}_\mathrm{s}^m(f_0) = \mathrm{Re}(\mathbf{P}_\mathrm{s}^r(f_0)) = \mathrm{Re}(\mathbf{K}_\mathrm{s}) - f_0^2 \cdot 4\pi^2 \mathbf{M}_\mathrm{s}.$$

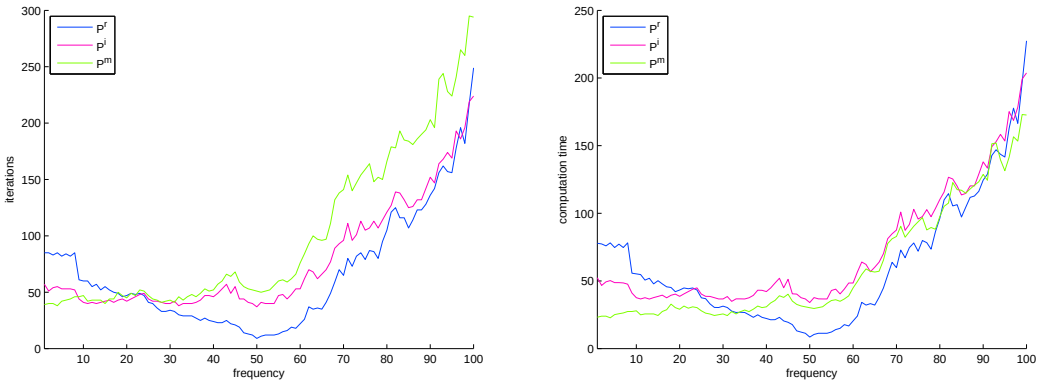
This preconditioner is also proposed by V. Mehrmann and C. Schröder in [14], although we derived this preconditioner independent from this article. We compare the computation time and numbers of iteration for all three shifted Laplace preconditioners. We choose IDR(8) and use the earlier defined initial search space $\mathcal{U}_0$ and since the type of extrapolation is not relevant we choose for the straightforward constant extrapolation, that is, $\mathbf{x}_0 = \mathbf{x}^{f-1}$, and we consider $f = 1, 2, \ldots, 100$ Hz and set $f_0 = 50$ Hz. The results per frequency are presented in Figure 7.13 and the total time and total number of iterations is tabulated in Table 7.9.

- Objective: performance of the three shifted Laplace preconditioners.
- Problem: the fluid and the structure problem.
- Frequency range: $f = 1, 2, \ldots 100$ Hz.
- Method: IDR(s) with $s = 8$, with $\mathcal{U}_0$ and $\mathbf{x}_0 = \mathbf{x}^{f-1}$.
- Preconditioner: $\mathbf{P}_s^r(f_0)$, $\mathbf{P}_s^i(f_0)$, $\mathbf{P}_s^m(f_0)$ with $f_0 = 50$ Hz.
- Results: Figure 7.13 and Table 7.9.

	$\mathbf{P}_\mathrm{s}^r(f_0)$	$\mathbf{P}_\mathrm{s}^i(f_0)$	$\mathbf{P}_\mathrm{s}^m(f_0)$
Time (s)	6 081	7 062	6 002
Iterations	6 643	7 739	9 960

Table 7.9: Computational results for the structure problem with the three preconditioners.

If we compare the numbers of iterations, we see that the modified preconditioner is the most expensive of the three. However, since the application time of this preconditioner is cheaper, it outperforms in terms of computation time the other two slightly on the given interval. Note that the modified preconditioner behaves per iteration very similar to the imaginary shifted preconditioner.



(a) iterations (b) computation time (s)

Figure 7.13: Computational results for the structure problem with the three preconditioners.

We would expect that for the real shifted preconditioner $\mathbf{P}_{\mathrm{s}}^{r}(f_0)$ the number of iterations for the frequency $f = f_0$ would reduce to 1, since in this case $(\mathbf{P}_{\mathrm{s}}^{r}(f_0))^{-1}\mathbf{A}_{\mathrm{s}}(f) = \mathbf{I}$. However, the number of iterations is equal to $s + 1 = 9$. This behaviour occurs in some cases when we use an initial search space $\mathcal{U}_0$, which has dimension s. If we do not use this space, the number of iterations does equal 1. The need of $s + 1$ iterations is caused by the fact that after s iterations the first minimisation step takes place. We can for now easily remedy this shortcoming by not using the search space for the computation of the solution to $f = f_0$ if we apply $\mathbf{P}_{*}^{r}(f_0)$, that is, we set $\mathcal{U} = \emptyset$ whenever $f = f_0$.

7.4.3 Updating the preconditioner

Since the computation of the LU factors is completed in a relative small amount of time, we are able to compute a new preconditioner every now and then without effecting the computation time unacceptably much. We have studied this strategy for the room problem with the conventional LU factorisation and have concluded that the total number of iterations reduces drastically for the shifted Laplace preconditioner with real shift, see Appendix B. This is due to the fact that a real shifted Laplace preconditioner is very efficient in a small range of frequencies, which can be seen for instance in Figure 7.13 on the interval $f \in [30, 80]$. The preconditioner with real shift performs the best in this interval, while outside this interval the other two preconditioners need equal or less computation time.

The question that arises immediately if we want to update the preconditioner is: at what point should we choose a new preconditioner and what shift f_0 should we choose for this new preconditioner? A first answer to this question is based on the ratio between the application time and the factorisation time of the preconditioner. For the fluid problem, we observe that for the shifted Laplace preconditioner with a real shift the time to factorise the preconditioner is on average 1.03 seconds, while 100 iterations with this preconditioner are done in approximately 2.73 seconds. This means that in computation time 38 iterations are comparable to the factorisation of a new real shifted preconditioner. The shifted Laplace preconditioner with imaginary shift is factorised in 2.54 seconds while 100 iterations are performed in 7.15 seconds, so 36 iterations are comparable with the factorisation of this preconditioner.

Since the computation time that is needed to construct and factorise the preconditioner is comparable with approximately 40 iterations, it follows that as soon as we need more than 40 iterations to compute an accurate solution, we should have computed a new preconditioner for that step. We therefore update the preconditioner in the first experiment as soon as we need more than 40 iterations. This is reasonable because, while the number of iterations per frequency behaves erratic, it lies within a rather small bandwidth. The value of the shift f_0 of the preconditioner is set equal to the upcoming frequency f, which means that for the next iteration we will need just 1 iteration for the preconditioner with a real shift.

We compare this updating strategy for the shifted Laplace preconditioners $\mathbf{P}_{\mathrm{f}}^{r}(f_0)$ and $\mathbf{P}_{\mathrm{f}}^{i}(f_0)$ with the cases where we have set f_0 equal to the centre of the interval and we do not update the preconditioners. We choose again for IDR(8), use the previous solutions for the initial search space and constant extrapolation. The results are given in Figure 7.14, where an 'o' indicates the moment that we computed a new preconditioner for this frequency.

- Objective: performance of updated shifted Laplace preconditioners.
- Problem: the fluid problem.
- Frequency range: $f = 1, 2, \ldots 500$ Hz.
- Method: IDR(s) with $s = 8$, with $\mathcal{U}_0$ and $\mathbf{x}_0 = \mathbf{x}^{f-1}$.
- Preconditioner: $\mathbf{P}_{s}^{r}(f_0)$, $\mathbf{P}_{s}^{i}(f_0)$, where f_0 is either updated or not.
- Results: Figure 7.14.

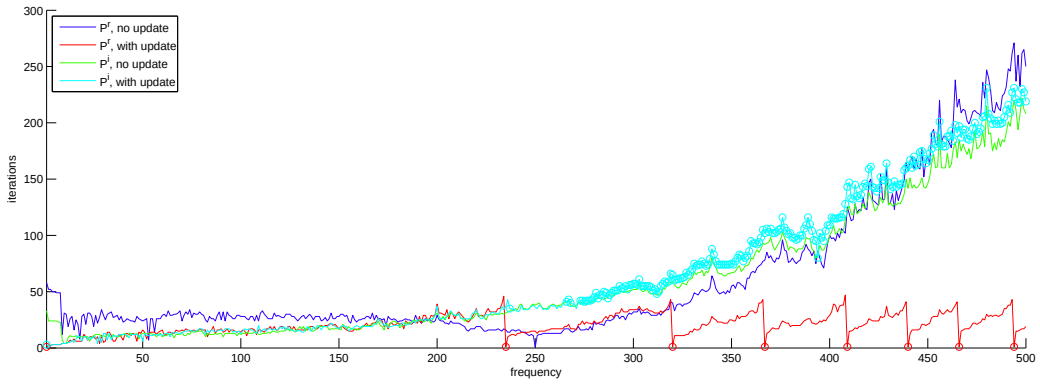


Figure 7.14: Number of iterations for the fluid problem, with/without update of the preconditioner.

For the imaginary shifted preconditioner $\mathbf{P}_{\mathrm{f}}^{i}(f_0)$, we see that there is almost no difference in numbers of iterations whether we update or not and hence it is not profitable to update this preconditioner. On the other hand, for $\mathbf{P}_{\mathrm{f}}^{r}(f_0)$ we see that the total number of iterations is very small where we update the preconditioner, while the number of iterations becomes very large for high frequencies that are not close to f_0. We also note that even for the frequencies near 500 Hz the updated real shifted Laplace preconditioner is still applicable to several frequencies before we compute a new preconditioner.

For the structure problem we do not take the updating of the shifted Laplace preconditioner with imaginary or modified real shift into consideration, since we do not expect any improvement compared to the unupdated preconditioner. The factorisation time for the shifted Laplace

preconditioner with real shift equals 36.4 seconds, while an iteration is done in on average 0.923 seconds and hence the computation of the LU factors is in time also equivalent to approximately 39 iterations. We therefore again update the preconditioner if more than 40 iterations are needed to compute the solution of a certain frequency. We consider a much smaller frequency interval, since the number of iterations increases drastically for higher frequencies and convergence does not occur for frequencies much higher than 100 Hz. We refer to Figure 7.15 for the results for the frequencies $f = 1, 2, \ldots, 100$ Hz.

- Objective: performance of updated shifted Laplace preconditioners.
- Problem: the structure problem.
- Frequency range: $f = 1, 2, \ldots 500$ Hz.
- Method: IDR(s) with $s = 8$, with $\mathcal{U}_0$ and $\mathbf{x}_0 = \mathbf{x}^{f-1}$.
- Preconditioner: $\mathbf{P}_s^r(f_0)$, where f_0 is either updated or not.
- Results: Figure 7.15.

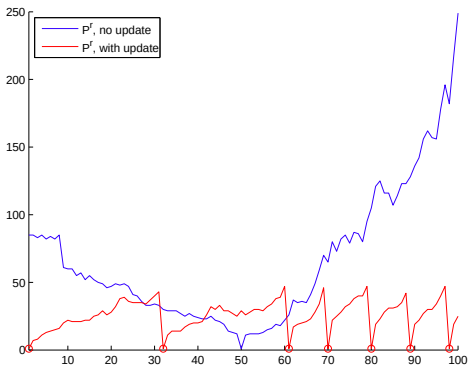


Figure 7.15: # iterations for the structure problem, with/without update of the preconditioner.

It follows that also for the structure problem the total number of iterations and the total computation time is much smaller if we apply and update the real shifted preconditioner. Note that for the fluid problem the first update was at approximately 230 Hz, while we need for the structure problem 6 updates for the first (and easiest) 100 frequencies. For frequencies above 60 Hz the preconditioner needs to be updated every 10 or less iterations and we expect that for higher frequencies we need to update even more frequently.

We compare the computation time for the preconditioner $\mathbf{P}_*^r(f_0)$ with and without update, see Table 7.10. In this table, we include the total time we need for the LU factorisation of the preconditioners. The reduction in computation time for the real shifted Laplace preconditioner that results from updating the shift is for both the fluid and the structure problem significant, and we expect a further improvement if we enlarge the interval and hence compute more frequencies. We are able to solve the fluid problem 2.7 times faster for the frequency range $f = 1, 2, \ldots, 500$ Hz, while the frequencies $f = 1, 2, \ldots, 100$ Hz of the structure problem are solved 2.3 times faster.

The choice of the update threshold q

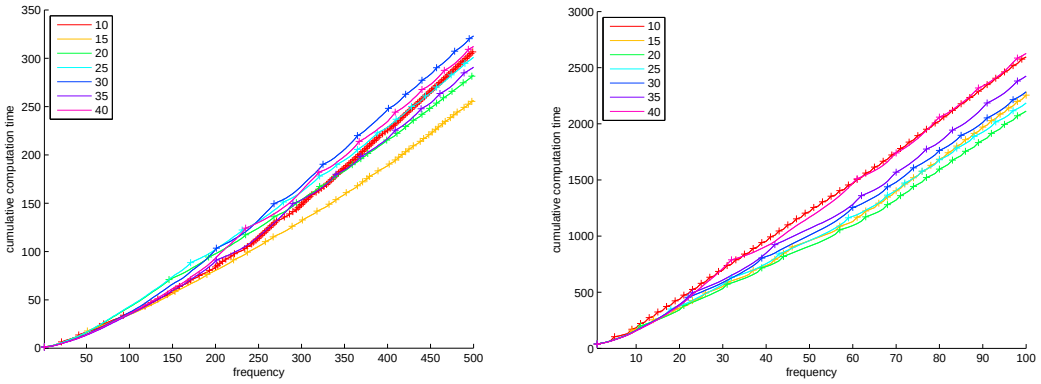
In the last set of experiments, we computed a new preconditioner if the number of iterations passed a threshold of $q = 40$. This is rather conservative choice and we could choose a smaller

	Without update	With update	Improvement
Fluid	858.8	313.6	63.5%
Structure	6 117	2 667	56.4%

Table 7.10: Computation time (s) for $\mathbf{P}_*^r(f_0)$ with/without updating the shift.

value for this threshold, since the computation time of the LU factors of the preconditioner is not relevant to a single frequency only, but also to the length of the frequency range we can apply it to. Therefore, we repeat the above experiment for $\mathbf{P}_*^r(f_0)$ with different values of q for the fluid and structure problem. The results are given in Figure 7.16, where the moment we update the preconditioner is marked with a '+'.

- Objective: investigation of the update threshold q.
- Problem: the fluid and structure problem.
- Frequency range: $f = 1, 2, \ldots 500$ Hz and $f = 1, 2, \ldots 100$ Hz.
- Method: IDR(s) with $s = 8$, with $\mathcal{U}_0$ and $\mathbf{x}_0 = \mathbf{x}^{f-1}$.
- Preconditioner: $\mathbf{P}_s^r(f_0)$ where the shift f_0 is updated.
- Results: Figure 7.16.



(a) fluid problem

(b) structure problem

Figure 7.16: Cumulative computation time (s) for different updated thresholds q.

We see in Figure 7.16 that for smaller frequencies the number of frequencies to which the preconditioner is applicable is larger than for higher frequencies, which implies that the number of iterations increases more rapidly for higher frequencies. We therefore expect that upward from a certain frequency the number of iterations will be larger than q for the frequency $f = f_0 + 1$ which means that we have to compute a new preconditioner every second iteration. This is of course undesirable, and we should therefore choose q not too small (at least not for high frequencies).

The described behaviour can be seen in Figure 7.16(a) for the choice $q = 10$. For all frequencies above 300 Hz, we update the preconditioner every second iteration, which leads to far too many updates and the corresponding curve is therefore much steeper than the others on the interval $[300, 500]$ Hz. Since for increasing frequencies the updated preconditioner is applicable to less

iterations, we expect that for every threshold we will reach a certain frequency after which this behaviour occurs. For instance, while the value $q = 15$ is still the best after 500 frequencies, we need to update the preconditioner far more often for the frequencies in $[300, 500]$ Hz than in the case where we choose $q = 20$ and we indeed see that the curve is steeper around 450 Hz. In Figure 7.16(b) we observe similar properties for the structure problem and while $q = 20$ is still the best choice after 100 frequencies, $q = 25$ or even $q = 30$ might be the best choice for frequencies up to 150 Hz.

Instead of fixing a threshold before we compute all frequencies, we could base the current choice of q on the frequency f or the number of frequencies that we are able to compute with a certain preconditioner. However, it is not clear how to effectively increase the value of q based on this information, while the expected improvement is very little. Therefore, we are satisfied with a fixed threshold of $q = 15$ for frequencies up to, say, 750 Hz for the fluid problem and for the structure problem we set $q = 25$ if we would compute all frequencies smaller than 150 Hz.

We conclude that choosing a proper threshold leads to a significant decrease in computation time. It is however very hard if not impossible to say in advance which choice is the best for a certain range of frequencies. Compared to the initial choice of $q = 40$, the reduction in computation time for the fluid (with $q = 15$) is 17.9% and for the structure (where we choose $q = 20$) equals 19.6%.

The choice of the shift parameter c

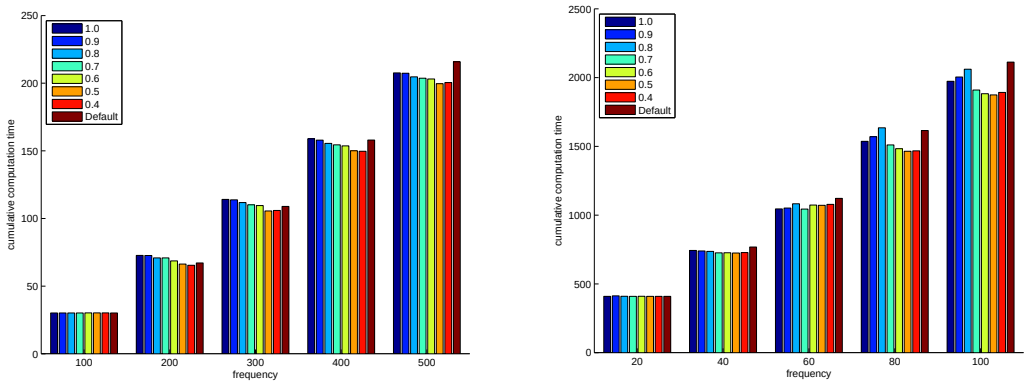
In the above, we fixed a shift for the preconditioner equal to $f_0 = f + 1$ as soon as the number of iterations for the frequency f was larger than the threshold. In order to make optimal use of the valley shape (which we observe for instance in Figure 7.13 around 50 Hz) we should probably choose f_0 a bit larger. In this experiment, we base our choice of f_0 on the number of frequencies that a shift is applied too, that is, we redefine f_0 as $f_0^{(\text{new})} = f + c \cdot (f - f_0^{(\text{old})})$ with $c \in (0, 1]$. We fix for the fluid problem a threshold $q = 15$ and set as initial shift $f_0 = 100$ Hz and for the structure problem we choose $q = 20$ and $f_0 = 20$ Hz, and we repeat the above experiment for values of c equal to $0.4, 0.5, \ldots, 1.0$. Some results concerning computation time are displayed in Figure 7.17. Note that the default choice uses the above initial values for f_0 instead of $f_0 = 1$ and hence there is already some improvement compared to the results displayed in Figure 7.16.

- Objective: investigation of the shift parameter c.
- Problem: the fluid and structure problem.
- Frequency range: $f = 1, 2, \ldots 500$ Hz and $f = 1, 2, \ldots 100$ Hz.
- Method: IDR(s) with $s = 8$, with $\mathcal{U}_0$ and $\mathbf{x}_0 = \mathbf{x}^{f-1}$.
- Preconditioner: $\mathbf{P}_s^r(f_0)$ where the shift f_0 is updated.
- Results: Figure 7.16.

We see that for both the fluid and the structure problem, the best choice appears to be $c = 0.5$. In this case, the reduction in total computation time (which is 208.4 seconds) is approximately 7.5% for the fluid and (with 2 097 seconds) equal to 11.3% for the structure problem.

Intermediate conclusions

Although the LU factorisation of the preconditioner is a demanding operation in terms of memory, it can be performed in a relatively short time (it equals in terms of computation time approximately 40 iterations). Therefore, the updating of the shifted Laplace preconditioners is a serious option. We draw the following conclusions concerning this updating:



(a) fluid problem

(b) structure problem

Figure 7.17: Cumulative computation time (s) for different choices of shift parameter c.

- Since the real shifted Laplace preconditioner $\mathbf{P}_*^r(f_0)$ behaves very well on a small interval around f_0 and much worse outside this interval, updating this preconditioner more than halves the computation time. Updating the imaginary shifted Laplace preconditioner $\mathbf{P}_*^i(f_0)$ does not improve the computation time and the precise shift of this preconditioner is of minor importance for the number of iterations for a given frequency. For the modified real shifted preconditioner $\mathbf{P}_s^m(f_0)$, it is not clear whether a valley is present around f_0 or not (see Figure 7.13(a)) and we have left this out of consideration for now.

- There is a lot of freedom in the choice of the moment that we update and in the choice of the shift f_0. The best results are obtained if we choose to update the preconditioner as soon as we need more than 15 iterations (for the fluid problem with frequencies $f = 1, 2, \ldots, 500$ Hz) or 20 iterations (for the structure problem with frequencies $f = 1, 2, \ldots, 100$ Hz) and set f_0 to be the computed frequency $f_0^{(\text{old})}$ plus half the number of frequencies $f > f_0^{(\text{old})}$ that we can apply the preconditioner with shift $f_0^{(\text{old})}$ to. The total reduction of computation time, compared to the standard choice (updating after 40 iterations and a shift equal to the next frequency) is for the fluid problem equal to 33.6 % and for the structure problem 21.4 %.

7.4.4 Other Krylov subspace methods

We consider IDR(s) with $s = 8, 16, 32$ and compare it to CGS, BiCGSTAB and GMRES. For the fluid experiment, we consider the real and imaginary shifted preconditioners $\mathbf{P}_{\text{f}}^r(250)$ and $\mathbf{P}_{\text{f}}^i(250)$ and $\mathbf{P}_{\text{f}}^r(f_0)$ where we update f_0. For this last preconditioner, we choose as initial shift $f_0 = 1$ Hz and if we need more than $q = 15$ (for the fluid) and $q = 20$ (for the structure) MATVECS, we set $f_0^{(\text{new})} = f + c \cdot (f - f_0^{(\text{old})})$, with $c = 0.5$. We apply these preconditioners to all frequencies $f = 1, 2, \ldots, 500$ Hz. We use for IDR(s) the standard initial search space $\mathcal{U}_0$ and use for all methods Lagrange extrapolation of s (for IDR(s) only) or 10 previous solutions. We have seen that for IDR(s) without an initial search space, this was the best choice and we expect similar results for other Krylov subspace methods. The number of MATVECS and computation time are per frequency displayed in Figure 7.18 and the totals are given in Table 7.11.

- Objective: performance of the preconditioners for several Krylov subspace methods.

- Problem: the fluid problem.
- Frequency range: $f = 1, 2, \ldots 500$ Hz.
- Method: IDR(s) with $s = 8, 16, 32$, CGS, BiCGSTAB and GMRES.
- Preconditioner: $\mathbf{P}_f^*(f_0)$ with a fixed frequency $f_0 = 250$ Hz and $\mathbf{P}_f^r(f_0)$ with initial shift $f_0 = 1$ Hz and updates $f_0 := f + 0.5 \cdot (f - f_0)$ if # MATVECS > 15 for $f - 1$.
- Results: Figure 7.18 and Table 7.11.

	Prec.	IDR(8)	IDR(16)	IDR(32)	CGS	BiCGSTAB	GMRES
Time (s)	$\mathbf{P}_{\mathrm{f}}^i(250)$	2 216	2 071	3 219	7 818	16 990	3 206
	$\mathbf{P}_{\mathrm{f}}^r(250)$	845	778	1 196	15 721	10 689	1 115
	$\mathbf{P}_{\mathrm{f}}^r(f_0)$	227	315	770	267	323	521
# MATVECS	$\mathbf{P}_{\mathrm{f}}^i(250)$	29 591	26 177	37 941	106 840	227 369	28 040
	$\mathbf{P}_{\mathrm{f}}^r(250)$	30 884	26 691	36 790	452 808	387 999	21 477
	$\mathbf{P}_{\mathrm{f}}^r(f_0)$	7 070	8 597	15 969	8 760	10 556	6 517

Table 7.11: Performance of different algorithms on the fluid problem.

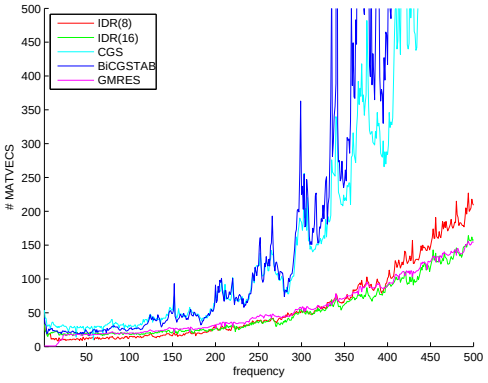
We do the above experiment for the structure problem also , but consider for this case also the preconditioner $\mathbf{P}_{\mathrm{s}}^m(f_0)$, without update. We set $f_0 = 50$ as shift for the preconditioners that we do not update. The number of MATVECS and computation time are given in Figure 7.19 and Table 7.12.

- Objective: performance of the preconditioners for several Krylov subspace methods.
- Problem: the structure problem.
- Frequency range: $f = 1, 2, \ldots 500$ Hz.
- Method: IDR(s) with $s = 8, 16, 32$, CGS, BiCGSTAB and GMRES.
- Preconditioner: $\mathbf{P}_s^*(f_0)$ with a fixed frequency $f_0 = 50$ Hz and $\mathbf{P}_s^r(f_0)$ with initial shift $f_0 = 1$ Hz and updates $f_0 := f + 0.5 \cdot (f - f_0)$ if # MATVECS > 20 for $f - 1$.
- Results: Figure 7.18 and Table 7.11.

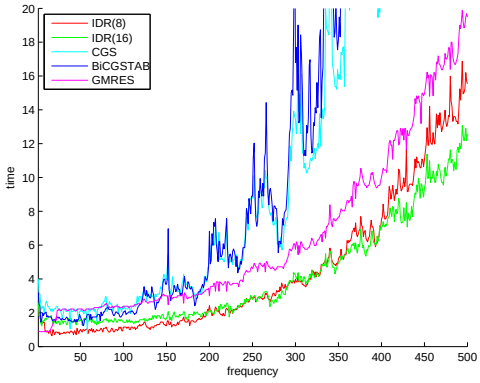
The relation between computation time and the number of MATVECS is almost linear for most methods, since the multiplications with the (preconditioned) system matrix, the MATVECS, are by far the most expensive operations. We see that this linearity property does not hold for GMRES and since this method uses long recurrences, this is no surprise.

The results of both the fluid and the structure problem need to be placed in the right perspective, since the proper accuracy is not obtained in some cases. We focus on the cases where we used a shifted Laplace preconditioner with a single fixed frequency first.

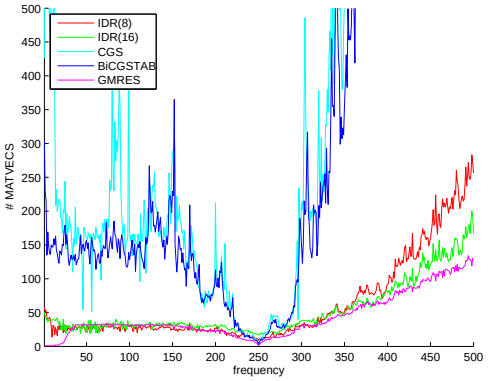
For the IDR(s) algorithms with $s = 16$ or $s = 32$, stagnation occurs for many frequencies. For IDR(16), the residual norms are in the worst case (that is, for high frequencies) still smaller than 10^{-5}, but for IDR(32) the results for all frequencies above 40 Hz for both problems are larger than 10^{-5} and for frequencies above 60 Hz so large that the solutions are too inaccurate and hence irrelevant. This inaccuracy and stagnation is presumably caused by the fact that the matrix $\mathbf{M}$, which contains the values $\mathbf{p}_k \cdot \mathbf{g}_j$ (see Algorithm 4), is ill-conditioned, which follows from the fact that for IDR(32) the estimated condition number in the 1-norm of this matrix, which is obtained by the LAPACK estimator, is larger than 10^{16} for all frequencies above 40 Hz.



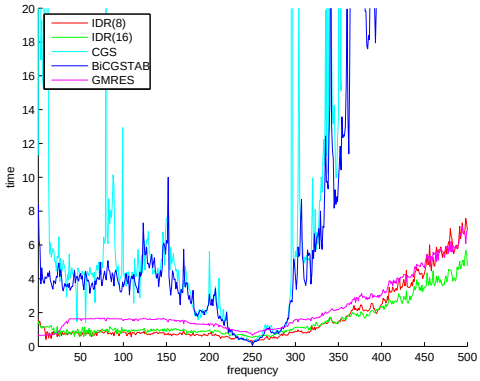
(a) $\mathbf{P}_{\mathrm{f}}^{i}(250)$, number of MATVECS



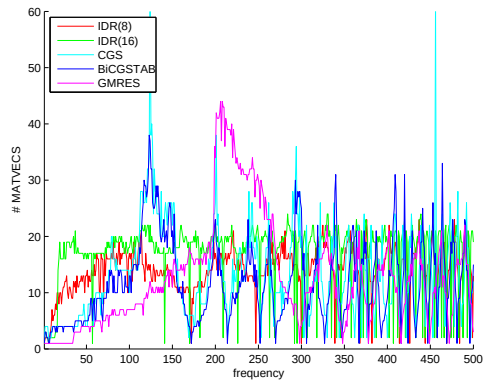
(b) $\mathbf{P}_{\mathrm{f}}^{i}(250)$, computation time (s)



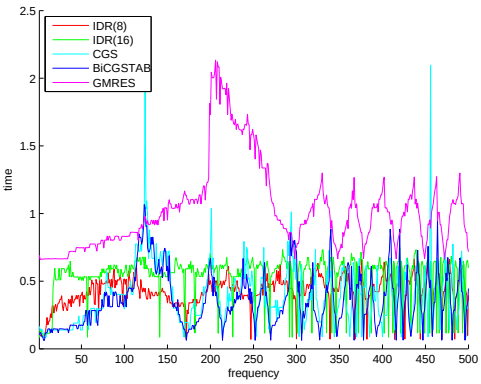
(c) $\mathbf{P}_{\mathrm{f}}^{r}(250)$, number of MATVECS



(d) $\mathbf{P}_{\mathrm{f}}^{r}(250)$, computation time (s)



(e) $\mathbf{P}_{\mathrm{f}}^{r}(f_0)$, number of MATVECS



(f) $\mathbf{P}_{\mathrm{f}}^{r}(f_0)$, computation time$(s)$

Figure 7.18: Performance of different algorithms on the fluid problem.

	Prec.	IDR(8)	IDR(16)	IDR(32)	CGS	BiCGSTAB	GMRES
Time (s)	$\mathbf{P}_s^i(50)$	7 061	7 063	10 567	32 642	83 252	7 963
	$\mathbf{P}_s^r(50)$	6 058	6 146	9 573	47 363	110 661	6 283
	$\mathbf{P}_s^m(50)$	4 871	5 275	7 884	45 088	78 987	5 554
	$\mathbf{P}_s^r(f_0)$	1 903	2 589	10 002	1 933	1 937	2 379
# MATVECS	$\mathbf{P}_s^i(50)$	7745	7198	9 854	24 546	90 407	5 360
	$\mathbf{P}_s^r(50)$	6 648	6 382	8 960	32 536	10 6181	4 312
	$\mathbf{P}_s^m(50)$	8 219	8 190	10 500	51 530	114 537	4 883
	$\mathbf{P}_s^r(f_0)$	1 616	1 740	8 018	1 544	1 559	1 289

Table 7.12: Performance of different algorithms on the structure problem.

The behaviour of CGS and BiCGSTAB are comparable for the fluid problem. The number of MATVECS increases drastically for frequencies above 300 Hz and for almost all of the higher frequencies the algorithm breaks down (one of the parameters becomes zero), stagnates or reaches the maximum of 1000 iterations, but in all cases the residual norm is still smaller than 10^{-5} and hence the solution is still rather accurate. For the structure problem, CGS breaks down for frequencies above 80 Hz (for $\mathbf{P}_s^i(50)$) or above 65 Hz (for $\mathbf{P}_s^r(50)$ and $\mathbf{P}_s^m(50)$), but in these cases the residual norm is smaller than 10^{-5}. BiCGSTAB needs more than 1000 iterations for the frequencies above 80 Hz, and for these frequencies the residual norm increases to values close to 10^6 and this means that BiCGSTAB is not an accurate method for the structure problem.

The algorithms IDR(8) and GMRES determine for any preconditioner for both problems the solution to all frequencies to a proper accuracy. In terms of computation time is IDR(16) the best choice.

If we consider the real shifted preconditioner $\mathbf{P}_*^r(f_0)$ where we use the described updating strategy, all the algorithms (except IDR(32)) behave very well. IDR(16) still stagnates, but the residuals are smaller than 10^{-6} for all frequencies. While for a single preconditioner CGS and BiCGSTAB behave much worse than IDR(s) and GMRES, they are comparable to these methods when we consider the updated $\mathbf{P}_*^r(f_0)$. This means that CGS and BiCGSTAB improve greatly and this improvement can be explained by the relative elongated valley that can be seen in Figures 7.18(c) and 7.19(c). At the right of the shift f_0, the number of MATVECS is very small and with the updating strategy we exploit this property.

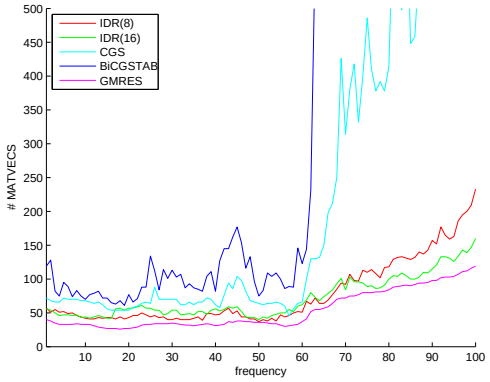
With the updating strategy of the real shifted preconditioner, IDR(8), CGS and BiCGSTAB are comparable both in computation time and number of MATVECS for the fluid and for the structure problem, see Tables 7.11 and 7.12.

The valley for IDR(s) around $f = f_0$ is very steep (see for instance Figure 7.18(e)), that is, there is just a small peak to 1 iteration, and for all other frequencies we always need $s + 1$ iterations. This means that the initial search space $\mathcal{U}_0$ again causes the algorithm to take $s + 1$ iterations while it needs maybe less iterations to obtain the proper accuracy. For small values of s this is no problem, but this might corrupt the performance of for instance IDR(16). Since in this case an initial guess based on Lagrange extrapolation of previous solutions is the best alternative we perform the above experiments for the fluid and structure problem without $\mathcal{U}_0$ and Lagrange extrapolation. The results are displayed in Figure 7.20 and Table 7.13.

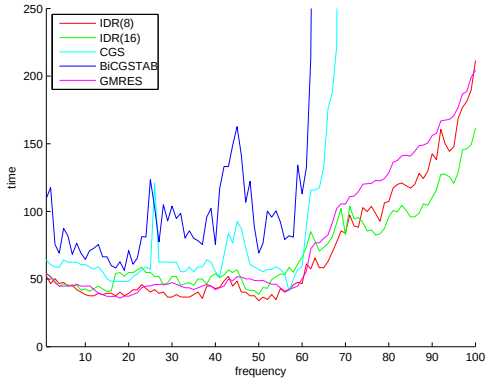
	Fluid		Structure	
	IDR(8)	IDR(16)	IDR(8)	IDR(16)
Time (s)	180	217	1 691	1 810
# MATVECS	5 433	6 191	1 460	1 473

Table 7.13: Performance of IDR(s) with $\mathbf{P}_*^r(f_0)$ and without $\mathcal{U}_0$.

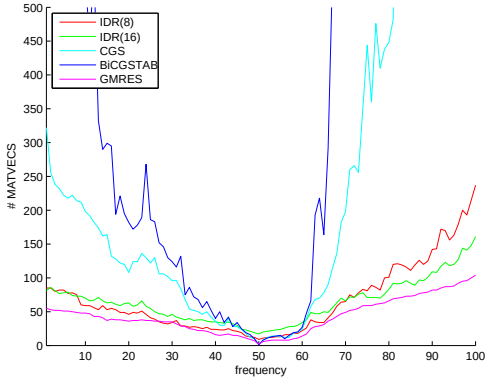
Without initial search space, the computation time and number of MATVECS is smaller compared to the same experiment with initial space. As we expect, the relative reduction for IDR(16) (which is 39% for the fluid problem) is higher compared to the reduction for IDR(8) (with 23% for the fluid problem) and since we update the preconditioner for the fluid problem much more often for high frequencies, the relative improvement for the fluid problem is also higher than for the structure problem (which equals 10% for IDR(16)). This means that with this strategy, which is in fact the same strategy we used for all other algorithms, IDR(8) and IDR(16) outperform CGS and BiCGSTAB slightly. Although the space $\mathcal{U}_0$ reduces the total number iterations if we



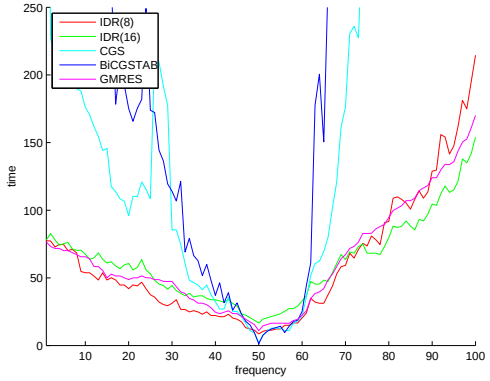
(a) $\mathbf{P}_{\mathrm{s}}^i(50)$, number of MATVECS



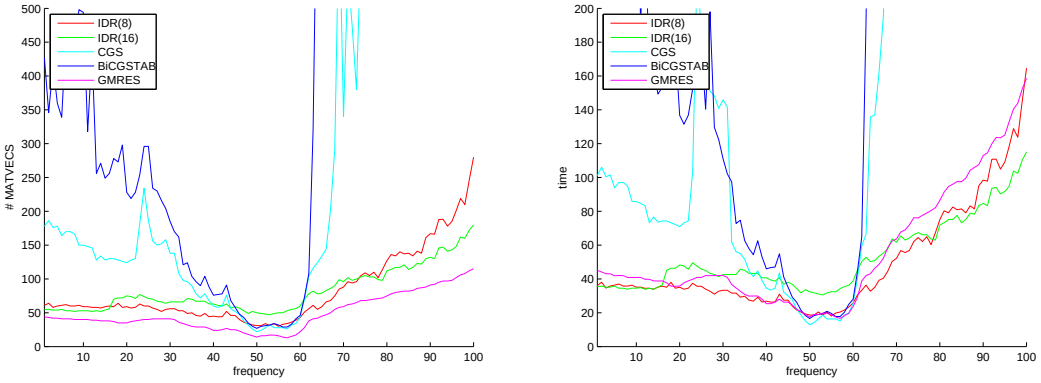
(b) $\mathbf{P}_{\mathrm{s}}^i(50)$, computation time ($s$)



(c) $\mathbf{P}_{\mathrm{s}}^r(50)$, number of MATVECS

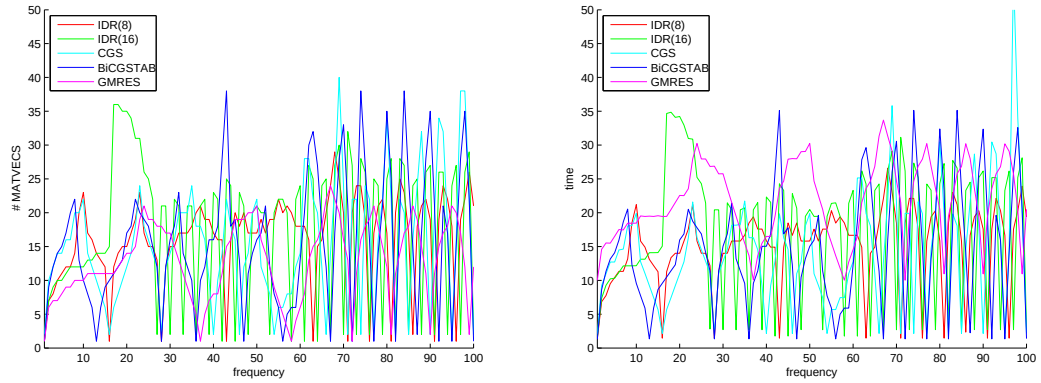


(d) $\mathbf{P}_{\mathrm{s}}^r(50)$, computation time ($s$)



(e) $\mathbf{P}_s^m(50)$, number of MATVECS

(f) $\mathbf{P}_s^m(50)$, computation time (s)



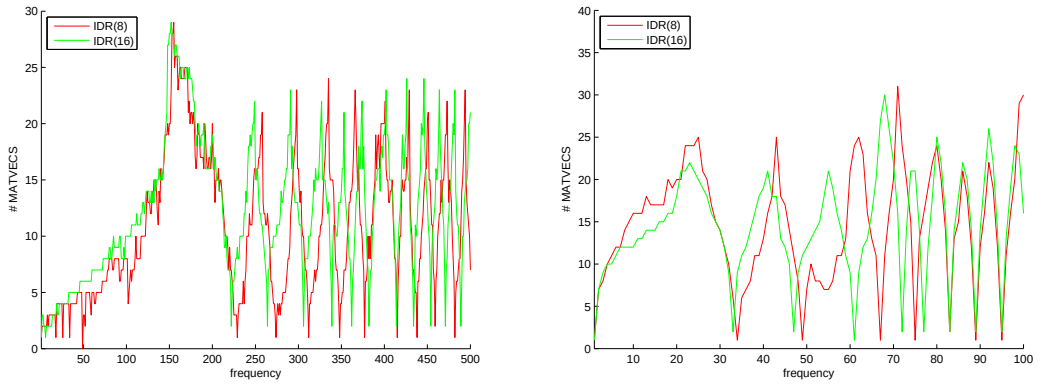
(g) $\mathbf{P}_s^r(f_0)$, number of MATVECS

(h) $\mathbf{P}_s^r(f_0)$, computation time (s)

Figure 7.19: Performance of different algorithms on the structure problem.

use a single preconditioner for a whole range of frequencies, it turns out that the performance of IDR(s) is better if we use no initial search space at all if we update our preconditioner. IDR(8) needs the least amount of computation time and hence is the best method for both the fluid and the structure problem.

We remark that CGS and BiCGSTAB have a small disadvantage compared to IDR(s): the convergence behaviour is more irregular and the 'walls of the valley' are very steep. This implies that these methods are very sensitive to the moment we compute a new preconditioner (based on q) and to the shift of this preconditioner (based on c). If we for instance take as shift a too high value, the number of iterations for the first few frequencies largely exceeds the threshold q. This can be seen for instance for BiCGSTAB in Figure 7.19(g). For $f = 42$ Hz we pass the threshold $q = 20$ and we therefore choose as new preconditioner $\mathbf{P}_s^r(47)$, after which we need 38 MATVECS for $f = 43$ Hz. For IDR(s) the numbers of iterations of subsequent frequencies are much more close together.



(a) fluid problem

(b) structure problem

Figure 7.20: Performance of IDR(s) with $\mathbf{P}_*^r(f_0)$ and without $\mathcal{U}_0$.

7.4.5 Conclusions

In this section, we investigated ways to improve the computation time for the fluid problem (7.4) and structure problem (7.5). We were able to solve the fluid problem for the frequencies $f = 1, 2, \ldots, 500$ Hz and for the structure problem for the frequencies $f = 1, 2, \ldots, 100$ Hz. The main conclusions based on the above experiments are given below.

- The different preconditioners $\mathbf{P}_*^i(f_0)$, $\mathbf{P}_*^r(f_0)$ and $\mathbf{P}_*^m(f_0)$ with a fixed shift f_0 equal to the centre of the interval, convert the system matrices so that the problems can be solved for a set of (relatively small) frequencies to a proper accuracy and within 1000 iterations.
- The solution to earlier obtained frequencies can be used effectively to reduce the computation time and number of iterations if we solve a sequence of problems. We can use the s nearest solution vectors by extrapolation them to improve the initial guess and as span for the initial search space $\mathcal{U}_0$. Both strategies (and their combination) reduce the total number of iterations that are needed for convergence for a sequence of systems.
- We exploit the fact that the real shifted Laplace preconditioner behaves very well around its shift f_0, by changing the shift of this preconditioner as soon as the number of iterations becomes too large. To make optimal use of this valley shape, we fix a shift f_0 a few iterations to the right. This approach leads to a serious reduction in computation time.
- In comparison to other well-known Krylov subspace methods, IDR(s) performs very well. We remark that the value of s should not be too small since then the number of iterations increases too much for high frequencies, while for large s, the method is unstable for both problems. We have seen that spanning an initial search space with previous solutions is not efficient if we use an updated real shifted Laplace preconditioner.

7.5 The complete car problem

We consider in this section the complete problem (7.3) and apply some of the most promising strategies that we developed in the last section to this problem. Since the structure part is the dominating and hardest part of the problem, we expect similar computational results for the

complete problem. Before we investigate the strategies, we study the accuracy of the solution to the problem.

7.5.1 Simultaneous computation of the structure and fluid problem

If we combine the structure and the fluid problem (and include the interaction term), the resulting (preconditioned) system matrix is badly scaled, as follows from the fact that the LAPACK estimated condition number in the 1-norm is larger than 10^{16}. This bad scaling property might lead to very inaccurate results for either the structure part or the fluid part of the solution, while the combined solution does satisfy the desired tolerance of 10^{-8}. We therefore compare the accuracy of the case where the fluid problem (7.4) and structure problem (7.5) are solved separately with the solution that we obtain by the simultaneous computation of the structure and fluid part, that is, by solving

$$\begin{pmatrix} \mathbf{A}_\mathrm{s}(f) & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_\mathrm{f}(f) \end{pmatrix} \begin{pmatrix} \mathbf{x}_\mathrm{s} \\ \mathbf{x}_\mathrm{f} \end{pmatrix} = \mathbf{A}_\mathrm{sf}(f) \begin{pmatrix} \mathbf{x}_\mathrm{s} \\ \mathbf{x}_\mathrm{f} \end{pmatrix} = \begin{pmatrix} \mathbf{b}_\mathrm{s} \\ \mathbf{b}_\mathrm{f} \end{pmatrix}. \tag{7.8}$$

Scaling of the preconditioned system matrix

We consider as a test the frequencies $f = 10$ Hz and $f = 100$ Hz for the structure and the fluid problem. The corresponding norms of the solutions that we obtain by IDR(8) with tol $= 10^{-8}$ are tabulated in Table 7.14. It follows that in both norms the solution to the structure problem is greater than the norms of the solution to the fluid problem. This suggests that the structure problem dominates the fluid problem and that if we solve (7.8), the fluid part might be inaccurate. Since the fluid problem needs far less iterations compared to the structure problem, we expect that inaccurate result will not occur.

	Frequency	1-norm	∞-norm
$\mathbf{x}_\mathrm{s}$	$f = 10$ Hz	13.5	$4.96 \cdot 10^{-3}$
	$f = 100$ Hz	25.8	0.284
$\mathbf{x}_\mathrm{f}$	$f = 10$ Hz	$1.94 \cdot 10^{-10}$	$1.22 \cdot 10^{-14}$
	$f = 100$ Hz	$8.79 \cdot 10^{-12}$	$9.67 \cdot 10^{-15}$

Table 7.14: Norms of individually computed solutions.

We investigate the accuracy of the solution to (7.8) and the solution to two modified versions of the problem where we use constant or diagonal scaling of the system matrix $\mathbf{A}_\mathrm{sf}(f)$, by comparing them to the separately obtained solutions. The application of the constant scaling includes the system matrix

$$\widetilde{\mathbf{A}}_\mathrm{sf}(f) = \begin{pmatrix} \mathbf{A}_\mathrm{s}(f) & \mathbf{0} \\ \mathbf{0} & \sigma \cdot \mathbf{A}_\mathrm{f}(f) \end{pmatrix},$$

with σ a scaling factor. The solution to the problem with the above system matrix $\tilde{\mathbf{x}}_\mathrm{sf}$ must be scaled back to obtain the desired solution:

$$\begin{pmatrix} \mathbf{x}_\mathrm{s} \\ \mathbf{x}_\mathrm{f} \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{x}}_\mathrm{s} \\ \sigma \cdot \tilde{\mathbf{x}}_\mathrm{f} \end{pmatrix}.$$

Since the stiffness matrix $\mathbf{K}$ has the largest values (even for large values of f), we consider the scaling factor $\sigma = \|\mathbf{K}_s\|_1 / \|\mathbf{K}_f\|_1 = 5.5673 \cdot 10^{-9}$.

For diagonal scaling, we divide each entry on a column of the system matrix with the corresponding element on the diagonal. That is, we scale the system matrix by right multiplication of the system matrix with its inverse diagonal elements and hence we consider the modified system matrix

$$\widehat{\mathbf{A}}_{\mathrm{sf}}(f) = \mathbf{A}_{\mathrm{sf}}(f)\mathbf{D}_{\mathbf{A}}(f)^{-1},$$

where $\mathbf{D}_{\mathbf{A}}(f)$ is a diagonal matrix with its non-zero entries equal to the diagonal entries of $\mathbf{A}_{\mathrm{sf}}(f)$. Again, the unscaled solution $\hat{\mathbf{x}}_{\mathrm{sf}}$ is scaled back by $\mathbf{x}_{\mathrm{sf}} = [\mathbf{D}_{\mathbf{A}}(f)]^{-1}\hat{\mathbf{x}}_{\mathrm{sf}}$. Note that the scaling of the system matrix is in fact a type of preconditioning and in this case we have that $\mathbf{P}_{\mathrm{R}} = \mathbf{D}_{\mathbf{A}}(f)$ in (4.10).

The condition numbers of the scaled system matrices are approximately 10^{12} and although this is very high (the reciprocal of this condition number is only 10^4 times the machine precision), it is a significant improvement compared to unscaled system matrix. Since the solutions to the problems with matrices $\mathbf{A}_{\mathrm{sf}}(f)$, $\widetilde{\mathbf{A}}_{\mathrm{sf}}(f)$ and $\widehat{\mathbf{A}}_{\mathrm{sf}}(f)$ ($f = 10$ Hz and $f = 100$ Hz) coincide with the case where we compute the structure and fluid solution separately for at least 6 decimal places, both types of scaling do not influence the accuracy and this suggest that the solutions to the unscaled problem are accurate too. Therefore, we do not use any type of scaling (which would in fact be less trivial for the system matrix where we include the damping term).

Including the damping term

The interaction term $\mathbf{D}_{\mathrm{sf}}$ changes the solution drastically (as we see by comparing the results in the Tables 7.15 and 7.14) and hence it is not usefull to use the separately obtained solutions to the fluid and structure parts for instance as initial guess for the complete problem.

	Frequency	1-norm	∞-norm
Structure	$f = 10$ Hz	$7.81 \cdot 10^{-6}$	$5.68 \cdot 10^{-10}$
	$f = 100$ Hz	$6.71 \cdot 10^{-7}$	$6.96 \cdot 10^{-10}$
Fluid	$f = 10$ Hz	44.6	$6.75 \cdot 10^{-3}$
	$f = 100$ Hz	6.71	$4.16 \cdot 10^{-2}$

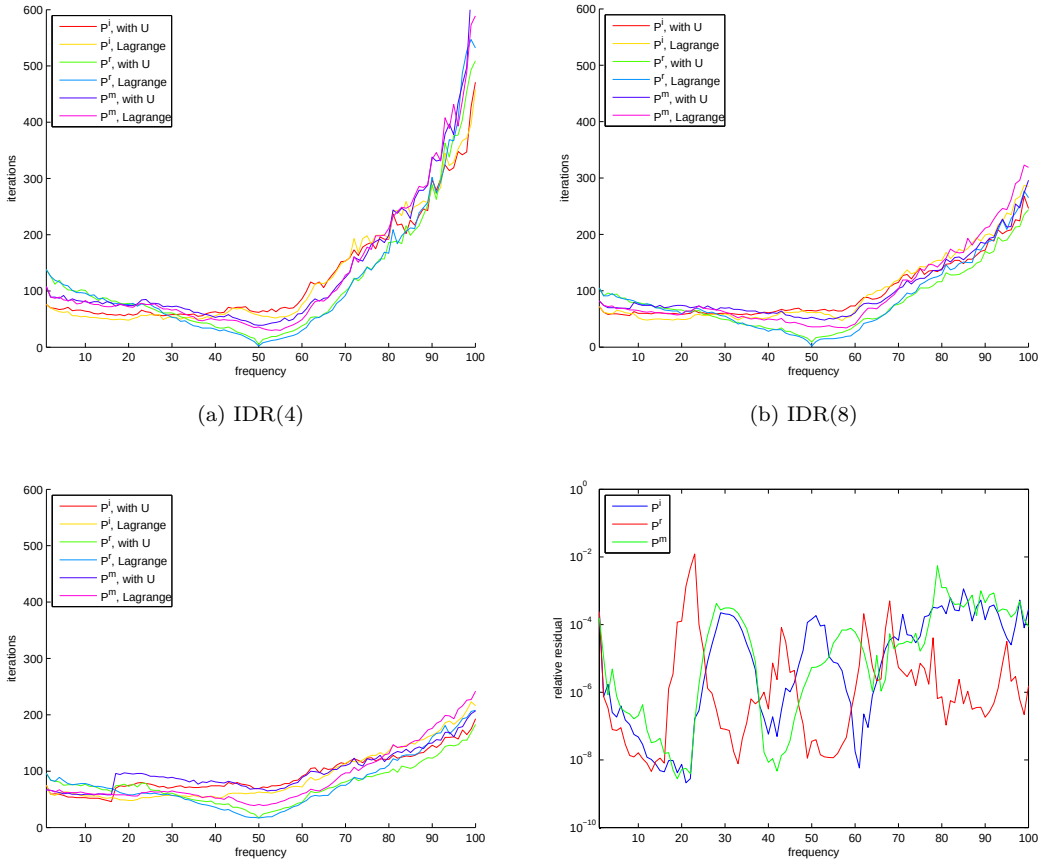
Table 7.15: Norms of the fluid part and the structure part of the complete solution.

7.5.2 Fixed shifted Laplace preconditioners

We consider for IDR(s) with $s = 4, 8, 16$ the three shifted Laplace preconditioners $\mathbf{P}^i(f_0)$, $\mathbf{P}^r(f_0)$ and $\mathbf{P}^m(f_0)$ with a fixed shift frequency $f_0 = 50$. We compare the results for two cases. The first case is with the span of the s previous solutions as initial search space $\mathcal{U}_0$ combined with the initial guess equal to the solution of the previous frequency, while in the second case we do not use an initial space $\mathcal{U}_0$ and we base the initial guess on the Lagrange extrapolation of the previous s solutions. We consider all frequencies $f = 1, 2, \ldots, 100$ Hz and display the numbers of iterations per frequency in Figure 7.21(a)-(c) and the total computation time and number of iterations in Table 7.16.

- Objective: performance of $\mathcal{U}_0$.
- Problem: the car problem.
- Frequency range: $f = 1, 2, \ldots 100$ Hz.

- Method: IDR(s) with $s = 4, 8, 16$, Lagrangian extrapolation of $\mathbf{x}^{f-i}$ $(i = 1, 2, \ldots, s)$.
- Preconditioner: $\mathbf{P}^r(f_0)$, $\mathbf{P}^i(f_0)$ and $\mathbf{P}^m(f_0)$ with $f_0 = 50$.
- Results: Figure 7.21 and Table 7.16.



(a) IDR(4)

(b) IDR(8)

(c) IDR(16)

(d) residual norms of IDR(16)

Figure 7.21: Number of iterations and residual norms for the car problem with/without $\mathcal{U}_0$.

For most of the higher frequencies and some smaller frequencies, the desired accuracy is not obtained, but for IDR(4) and IDR(8) the true residual norm is still smaller than 10^{-5}. For IDR(16), a satisfactory residual norm is obtained for less than half of the frequencies, which can be seen in Figure 7.21(d), where we display per frequency the norm of the residual that we obtain with IDR(16). Apparently, some frequencies are much more difficult than others and the difference in residual norms can be very large for frequencies close to each other (consider for instance $f = 22, \ldots, 25$ Hz). We also note that there is some resemblance in the curves that correspond to the preconditioners $\mathbf{P}^i$ and $\mathbf{P}^m$. The bad convergence is again caused by stagnation of IDR(s) due to the fact that the matrix $\mathbf{M}$ is ill-conditioned.

In terms of computation time and number of iterations, we see from Table 7.16 that IDR(8) and IDR(16) are comparable and always outperform IDR(4). Since IDR(16) does not lead to satisfactory results, it follows that IDR(8) is the best choice for all cases. The difference in computation

		with $\mathcal{U}_0$, constant extrapolation			no $\mathcal{U}_0$, Lagrange extrapolation		
	Prec.	IDR(4)	IDR(8)	IDR(16)	IDR(4)	IDR(8)	IDR(16)
Time (s)	$\mathbf{P}^i(50)$	21 573	16 577	16 477	21 304	16 958	16 108
	$\mathbf{P}^r(50)$	19 930	13 845	13 369	20 111	14 354	13 794
	$\mathbf{P}^m(50)$	8 488	6 399	6 948	8 532	6 385	6 266
Iterations	$\mathbf{P}^i(50)$	12 665	9 725	9 373	12 513	9 955	9 140
	$\mathbf{P}^r(50)$	11 713	8 062	7 567	11 809	8 427	7 824
	$\mathbf{P}^m(50)$	13 848	9 946	9 929	13 408	9 864	8 887

Table 7.16: Computational results for the car problem with/without $\mathcal{U}_0$.

time in on the one hand IDR(4) and on the other hand IDR(8) and IDR(16) is mainly caused by the much less iterations that are needed in the last case for high frequencies, which can be seen from Figure 7.21(a)-(c).

It does not make a significant difference whether we use the initial search space $\mathcal{U}_0$ or not. The total number of iterations is for all three preconditioners almost the same, as we see in Table 7.16. The corresponding curves in Figure 7.21(a) almost coincide for all preconditioners and in 7.21(b)-(c) we see that using the space $\mathcal{U}_0$ leads to less iterations for $71, \ldots, 100$ Hz and more iterations for the frequencies $s+1, \ldots, 70$ Hz. It is remarkable that at the moment that we apply the initial search space, that is, at $f = s+1$, a significant increase in the number of iterations occurs.

Although the number of iterations is much higher for the modified real shifted Laplace preconditioner, this preconditioner outperforms the others, since the factorisation is 9 times less time consuming while the application time is reduced by a factor of 2,5. Therefore, the modified preconditioner $\mathbf{P}^m(f_0)$ is in terms of computation time the best preconditioner.

7.5.3 Updating the preconditioners

While the preconditioners $\mathbf{P}^m(f_0)$ and $\mathbf{P}^i(f_0)$ show some similarity in the number of iterations per frequency, a small valley is apparent near 50 Hz for $\mathbf{P}^m(f_0)$, as we see in Figure 7.21(a)-(c). Therefore, we take also the updating of the modified real shifted Laplace preconditioner $\mathbf{P}^m(f_0)$ into consideration. We study the methods IDR(8), IDR(16), CGS, BICGSTAB and GMRES and consider the preconditioners $\mathbf{P}^r(f_0)$ and $\mathbf{P}^m(f_0)$.

The time that is needed for the factorisation of $\mathbf{P}^r(f_0)$ is approximately 146.9 seconds, and since one iteration of IDR(s) takes more or less 1.76 seconds, 83 iterations are comparable to the factorisation. We performed some experiments in order to choose proper values for the threshold q and update parameter c and it turns out that these values should be chosen equivalent to the structure problem, that is, a threshold of $q = 45 \approx 0.55 \cdot 83$ leads to the best results for all algorithms. We set as update parameter $c = 0.5$ for all algorithms but GMRES, since for this algorithm $c = 0.3$ leads to much better results.

The decomposition of $\mathbf{P}^m(f_0)$ takes only 16.1 seconds and a single iteration of IDR(s) is done in 0.705 seconds, which implies that 23 iterations are in computation time equal to the factorisation of the preconditioner. The conventional choice of the threshold would again be somewhere around 0.5 or 0.6 times 23, but this is not a proper choice if we use the modified preconditioner since

the system matrix of the preconditioned system does not reduce to an (approximation of) the identity matrix for $f = f_0$ and the minimum number of iterations equals approximately 40. If we want to apply the strategy similar to the case with the real shifted preconditioner, we need to choose q much larger. Experiments show that a choice of $q = 70$ is the best. For this value the preconditioner is still applicable to several frequencies, also for frequencies close to 100 Hz. The values of c are the same, that is, for IDR(s), CGS and BiCGSTAB $c = 0.5$, and for GMRES $c = 0.3$.

We use the above algorithms and settings, and use Lagrangian extrapolation of previous solutions for the initial guess to solve the preconditioned car problem with preconditioners $\mathbf{P}^r(f_0)$ and $\mathbf{P}^m(f_0)$. The results are displayed in Figure 7.17 and Table 7.22. An 'o' again indicates the moment where f_0 is updated and hence a new factorisation is performed.

- Objective: performance of the preconditioners for several Krylov subspace methods.
- Problem: the car problem.
- Frequency range: $f = 1, 2, \ldots 100$ Hz.
- Method: IDR(s) with $s = 8, 16$, CGS, BiCGSTAB and GMRES.
- Preconditioner: $\mathbf{P}^r(f_0)$ and $\mathbf{P}^m(f_0)$ with initial shift $f_0 = 1$ and updates $f_0 := f + c \cdot (f - f_0)$ if # MATVECS $> q$ for $f - 1$.
- Results: Figure 7.22 and Table 7.17.

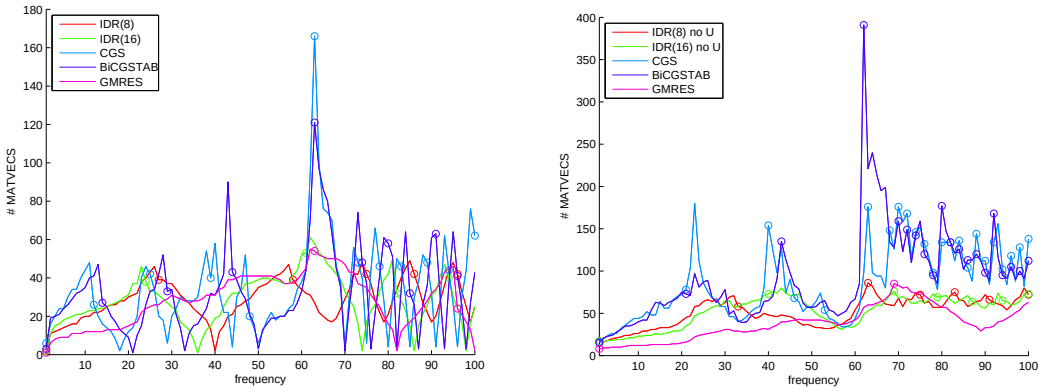
	Prec.	IDR(8)	IDR(16)	CGS	BiCGSTAB	GMRES
Time (s)	$\mathbf{P}^r(f_0)$	5 978	5 674	8 751	7 840	6 693
	$\mathbf{P}^m(f_0)$	3 686	3 806	7 396	7 289	4 344
# MATVECS	$\mathbf{P}^r(f_0)$	2 857	2 876	3 350	3 251	2 823
	$\mathbf{P}^m(f_0)$	5 816	5 731	9 142	9 283	3 820
# updates	$\mathbf{P}^r(f_0)$	6	5	12	10	3
	$\mathbf{P}^m(f_0)$	5	4	17	16	2

Table 7.17: Performance of different methods on the complete car problem.

Concerning the accuracy we can say that CGS and GMRES obtain a solution that corresponds to a residual norm smaller than tol. For BiCGSTAB, breakdown occurs for some frequencies and IDR(s) stagnates also for a few frequencies, but in these cases the residual norm is still smaller than 10^{-6}. For these frequencies, the system is most likely close to singular.

The moments that we update the shift of the preconditioners f_0 and hence compute new LU factors, indicated by an 'o' in Figure 7.22, are almost always at the right frequencies: they are very close to a peak in the number of MATVECS. We also note that for both CGS and BiCGSTAB, of which the results have some similarity, we update more often than for IDR(s) and GMRES. The peaks close to the frequencies where we update are also much higher. This is not caused by improper values for q and c (which we investigated), but these peaks are inherent to these methods: for two subsequent frequencies, the number of MATVECS for convergence can be very distant from each other. The combination of many updates and high peaks in the number of MATVECS results in a relatively bad performance of both CGS and BiCGSTAB.

GMRES is the best method if we would compute frequencies smaller than 50 Hz. Since for higher frequencies the number of iterations that GMRES needs for convergence increases and GMRES uses long recurrences, it becomes much worse for these frequencies and it is outperformed by



(a) $\mathbf{P}^r(f_0)$ (b) $\mathbf{P}^m(f_0)$

Figure 7.22: Performance of different methods on the complete car problem.

IDR(s).

It follows from Figure 7.22 that the modified shifted Laplace preconditioner $\mathbf{P}^m(f_0)$ does cause a valley shape in the number of MATVECS around $f = f_0$ and although much higher numbers of MATVECS are needed for convergence, it outperforms the real shifted Laplace preconditioner $\mathbf{P}^r(f_0)$ for all methods in terms of computation time.

7.5.4 Solution to the large car problem

In addition to the car model with $n = 192\,184$ unknowns, a model with a structure part of $n_s = 478\,788$ unknowns and hence a total of $n = 495\,151$ unknowns is available. With the techniques we have developed in the above sections, we are able to solve this much larger problem. Similar to the experiment of section 7.5.3, we consider IDR(8), CGS, BiCGSTAB and GMRES. We do not take IDR(16) into consideration, since stagnation occurs and the residual norms are too large for almost all frequencies. As preconditioner, we choose the modified shifted Laplace preconditioner $\mathbf{P}^m(f_0)$ and we update the shift f_0 if we pass a threshold of $q = 90$ to the new shift $f_0^{(\text{new})} = f + c \cdot (f - f_0^{(\text{old})})$ with $c = 0.5$, except for GMRES where we again choose $c = 0.3$. The number of MATVECS per iteration is displayed in Figure 7.23 and the totals of computation time and numbers of MATVECS are tabulated in Table 7.18.

- Objective: performance of the preconditioners for several Krylov subspace methods.
- Problem: the large car problem.
- Frequency range: $f = 1, 2, \ldots 100$ Hz.
- Method: IDR(8), CGS, BiCGSTAB and GMRES.
- Preconditioner: $\mathbf{P}^m(f_0)$ with initial shift $f_0 = 1$ and updates $f_0 := f + c \cdot (f - f_0)$ if # MATVECS > 90 for $f - 1$.
- Results: Figure 7.23 and Table 7.18.

The solutions to this car problem are less accurate then the solutions to the equivalent but smaller problem. With GMRES, the residual norms are always smaller than 10^{-7}, while for CGS these norms are not larger than 10^{-6}. The residual norms that result from using IDR(8) or BiCGSTAB

	IDR(8)	CGS	BiCGSTAB	GMRES
Time (s)	8 220	15 063	16 194	13 584
# MATVECS	6 427	9 674	11 029	4 824
# updates	4	23	23	2

Table 7.18: Performance of different methods on the large car problem.

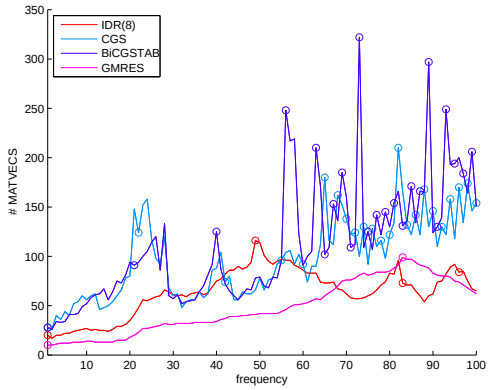


Figure 7.23: Performance of different methods on the large car problem.

are in the worst case close to 10^{-5}.

The computation time increases significantly, while the increase in the number of MATVECS is relatively small. We observe for the larger car problem very similar behaviour of all methods to the smaller problem. The curves of CGS and BiCGSTAB in Figure 7.23 are close together and extreme jumps in the number of MATVECS above 50 Hz are also present. In addition, these methods need far more updates than IDR(8) and GMRES, which behave much more stable. In terms of computation time is IDR(8) the best method, but due to its lack of accuracy for higher frequencies, GMRES is probably a better candidate, although it needs almost twice as much computation time.

7.5.5 Conclusions

We applied in this section the techniques that we developed for the fluid and structure problem to the complete car problem (7.3). We mainly focused on the problem with $n_s = 175\,821$ and $n = 192\,184$, but we were also able to solve the much larger car problem with a total of $n = 495\,151$. Based on the experiments, we draw the conclusions:

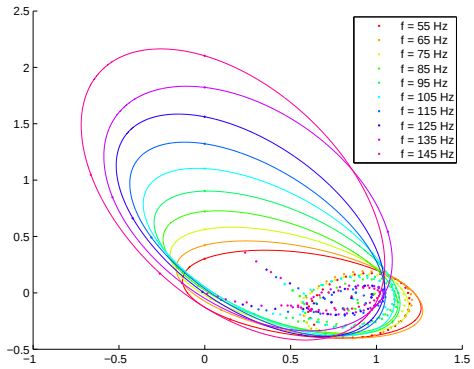
- The car problem is dominated by the structure problem, which is much larger than the fluid. The structure problem needs always more iterations than the fluid problem and the separate fluid and structure part solutions to the complete car problem are therefore accurate.
- The smaller car problem is only slightly larger than the structure problem, but much harder to solve. The car problem needs (with the same preconditioner) approximately 3 times more MATVECS compared to the structure problem and only GMRES is able to obtain a relative residual with a norm smaller than 10^{-8}.

- Updating the shifted Laplace preconditioners $\mathbf{P}^r(f_0)$ and $\mathbf{P}^m(f_0)$ is again the best way to solve this problem. The threshold q and shift parameter c that have the best performance for the car problem are equivalent with the best choices for the structure problem, except for GMRES where the shift for the new preconditioner needs to be much smaller.

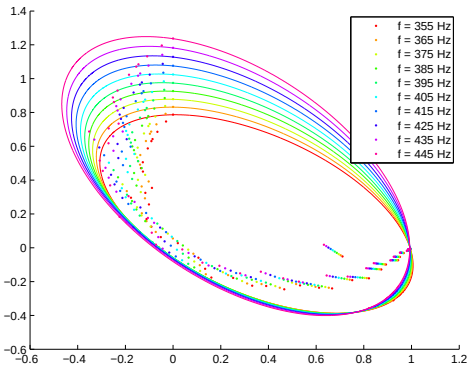
- Although updating the real shifted preconditioner $\mathbf{P}^r(f_0)$ reduces the number of iterations to much lower values compare to the updated modified preconditioner $\mathbf{P}^m(f_0)$, this latter preconditioner outperforms the real shifted version, especially for IDR(8) and GMRES.

7.6 Using eigenvalue information for reduction

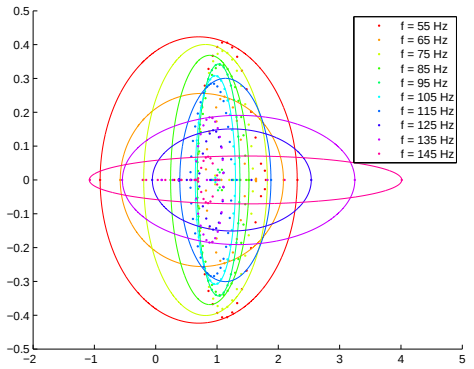
Instead of using solution vectors, we could use spectral information of the system matrix. The computation of the Ritz values, that are obtained by the relation (6.11) in IDR(s), does not influence the convergence behaviour and the increase in the computation time is negligible, since the number of iterations is much smaller than the size of the system matrices. We use for the solution to a frequency f the Ritz values of the system matrix of $f - 1$.



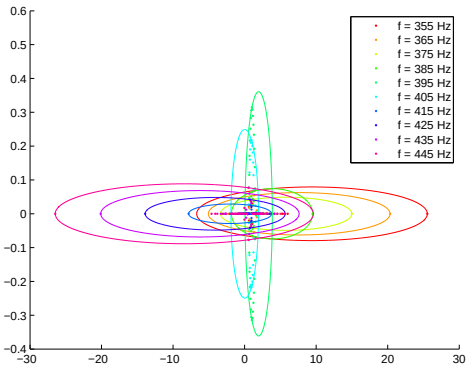
(a) $\mathbf{P}_{\mathrm{f}}^i(f_0)$, $f_0 = 100$



(b) $\mathbf{P}_{\mathrm{f}}^i(f_0)$, $f_0 = 400$



(c) $\mathbf{P}_{\mathrm{f}}^r(f_0)$, $f_0 = 100$



(d) $\mathbf{P}_{\mathrm{f}}^r(f_0)$, $f_0 = 400$

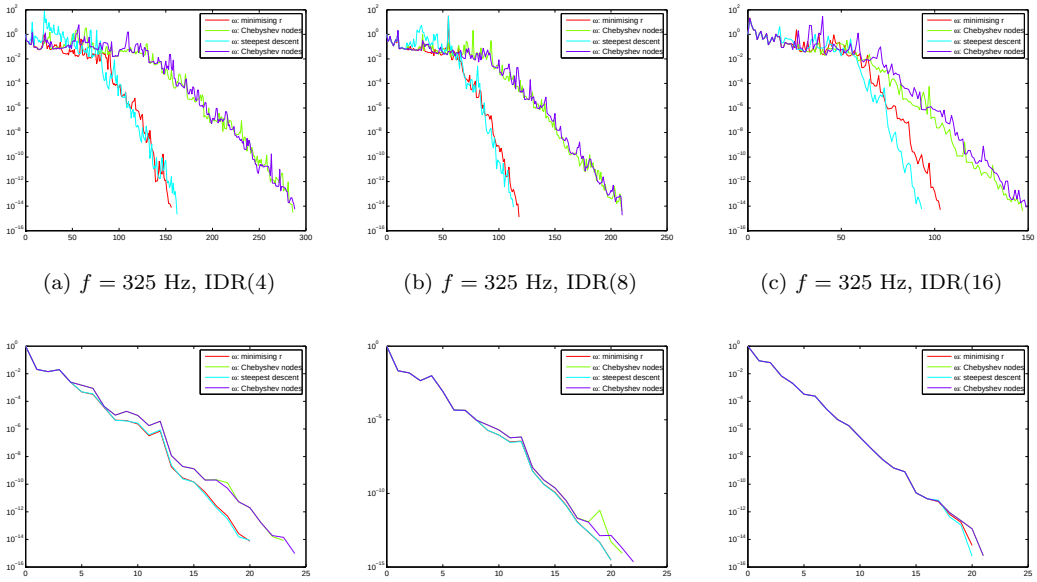
Figure 7.24: Ritz values and their minimal enclosing ellipses for the fluid problem.

We could directly use the reciprocals of the Ritz values for ω_j, as was done in for instance [22], but we choose to consider the reciprocals of ψ Chebyshev nodes that follow from the Chebyshev polynomial on the ellipse of minimal area that encloses all the Ritz values.

In Figure 7.24, the minimal enclosing ellipses that correspond to the Ritz values for the preconditioned fluid problem are displayed. As preconditioners we choose $\mathbf{P}_{\mathrm{f}}^{i}(f_0)$ and $\mathbf{P}_{\mathrm{f}}^{r}(f_0)$ with $f_0 = 100, 400$ Hz and we consider the frequencies $f_0 \pm k$ with $k = 5, 15, \ldots, 45$ Hz.

Note that for the real shifted preconditioner the ellipses are very elongated, especially for the case with $f_0 = 400$ Hz. We see from Figure 7.24 that the ellipses are close together, while the differences in frequencies is equal to 10 Hz. The resemblance is particularly present for the preconditioner with imaginary shift. This implies that the Chebyshev nodes that correspond to the problem with a frequency equal to f accurately approximate the Chebyshev nodes of the problem with $f - 1$ (if f_0 is held fixed).

For the fluid problem, we compare the convergence of IDR(s) for the frequencies $f = 325, 395$ Hz by taking the residual norms per iteration into consideration. We use as preconditioners $\mathbf{P}_{\mathrm{f}}^{r}(f_0)$ (see Figure 7.25) and $\mathbf{P}_{\mathrm{f}}^{i}(f_0)$ (see Figure 7.26) with $f_0 = 400$ Hz. First, we consider the two earlier described choices for ω_j in IDR(s), that is, the minimisation strategy of the first residual with respect to ω_j: $\omega_j = (\mathbf{A}\mathbf{v}_i \cdot \mathbf{v}_i)/(\mathbf{A}\mathbf{v}_i \cdot \mathbf{A}\mathbf{v}_i)$ (in red) and the method of steepest descent $\omega_j = (\mathbf{v}_i \cdot \mathbf{v}_i)/(\mathbf{v}_i \cdot \mathbf{A}\mathbf{v}_i)$ (in cyan). Second, we consider for ω_j the reciprocals of $\psi = 10$ Chebyshev nodes that result from the Ritz values that are obtained for the frequency $f - 1$, where these Chebyshev nodes are used cyclically as the IDR iterations proceed, except for the first s iterations, where we choose as always $\omega_0 = 1$. The choices for ω_j for $f-1$ are again $\omega_j = (\mathbf{A}\mathbf{v}_i \cdot \mathbf{v}_i)/(\mathbf{A}\mathbf{v}_i \cdot \mathbf{A}\mathbf{v}_i)$ (in green) and $\omega_j = (\mathbf{v}_i \cdot \mathbf{v}_i)/(\mathbf{v}_i \cdot \mathbf{A}\mathbf{v}_i)$ (in purple).

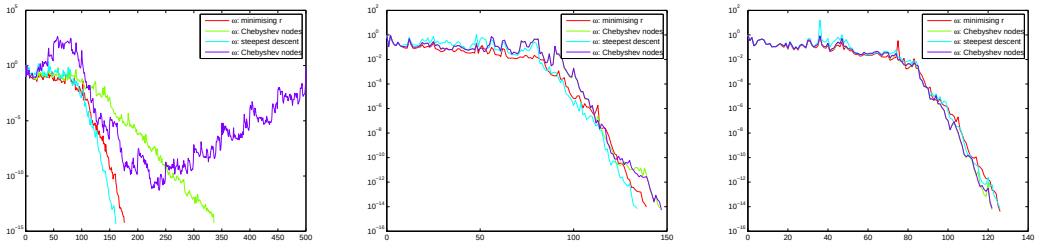


(a) $f = 325$ Hz, IDR(4) (b) $f = 325$ Hz, IDR(8) (c) $f = 325$ Hz, IDR(16)

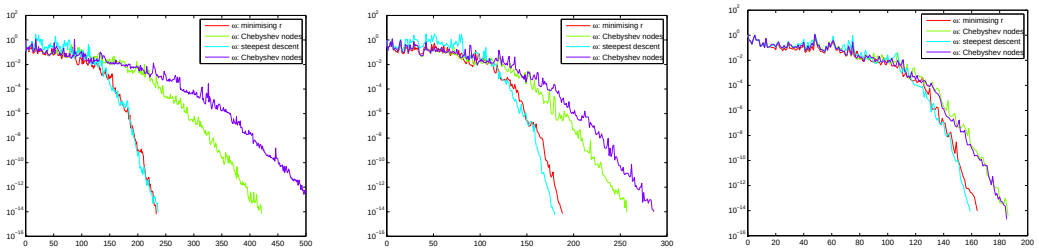
(d) $f = 395$ Hz, IDR(4) (e) $f = 395$ Hz, IDR(8) (f) $f = 395$ Hz, IDR(16)

Figure 7.25: Residual norms for the fluid problem with $\mathbf{P}_{\mathrm{f}}^{r}(f_0)$ and $f_0 = 400$.

It is remarkable that the method of steepest descent for the ω_j is at least as good as the minimisation strategy. For the cases where we use Chebyshev nodes, we see that the curves are mutually very similar for most cases, which can be explained by the fact that we compute only k Ritz values with $k \ll n$ and these Ritz values are in this case often very similar for different choices of ω_j. For the cases where there is a significant difference in Chebyshev nodes (see Figure 7.26(a),(d)), we see that the convergence behaviour for the case based on the Chebyshev nodes that follow from the steepest descent method for ω_j, is much worse. This is surprising since we have seen in section 6.4.2 that in this case the Ritz values better approximate the eigenvalues.



(a) $f = 325$ Hz, IDR(4) (b) $f = 325$ Hz, IDR(8) (c) $f = 325$ Hz, IDR(16)

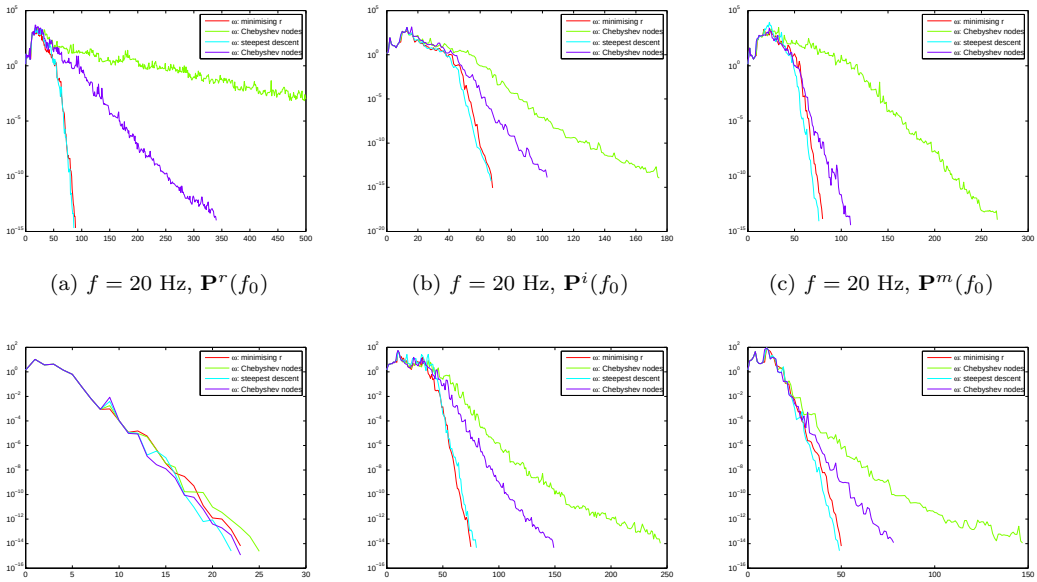


(d) $f = 395$ Hz, IDR(4) (e) $f = 395$ Hz, IDR(8) (f) $f = 395$ Hz, IDR(16)

Figure 7.26: Residual norms for the fluid problem with $\mathbf{P}_{\mathrm{f}}^{i}(f_0)$ and $f_0 = 400$.

We see from these figures also that if we choose for ω_j the reciprocals of Chebyshev nodes, the performance is never better than the choices based on $\mathbf{A}\mathbf{v}_i$ and $\mathbf{v}_i$, which implies that for this problem the Chebyshev nodes strategy seems to work not well. For increasing s for IDR(s), the curves are closer to each other, but this is due to the fact that the degree of the polynomial Ω_j is much lower for higher values of s.

We study a similar approach for the complete car problem, but note that for this case the resemblance for the Chebyshev nodes is much smaller. While most of the eigenvalues are clustered, a few eigenvalues are very distant from this cluster. This results in very elongated minimal enclosing ellipses that are differently orientated, while the clusters of eigenvalues are more or less the same. We consider all three shifted Laplace preconditioners with a fixed shift of $f_0 = 50$ Hz and determine the residual norms for $f = 20, 48$ Hz after each iteration. We do this for IDR(8) and give the results in Figure 7.27, where the choices for ω_j in IDR(s) are based on minimising a residual (in red) or steepest descent (in cyan) are compared with the choice based on the Chebyshev nodes and Ritz values that are obtained for $f - 1$ with minimising a residual (in green) or steepest descent (in purple).



(a) $f = 20$ Hz, $\mathbf{P}^r(f_0)$ (b) $f = 20$ Hz, $\mathbf{P}^i(f_0)$ (c) $f = 20$ Hz, $\mathbf{P}^m(f_0)$

(d) $f = 48$ Hz, $\mathbf{P}^r(f_0)$ (e) $f = 48$ Hz, $\mathbf{P}^i(f_0)$ (f) $f = 48$ Hz, $\mathbf{P}^m(f_0)$

Figure 7.27: Residual norms for the car problem with IDR(8) and $f_0 = 50$.

We see from this figure that also for the car problem the method of steepest descent for the ω_j is as good as the minimisation strategy. For the cases where we use Chebyshev nodes, we see that the curves are more distinct (except in Figure 7.27(d), where only ± 25 iterations are carried out). In contrast to the fluid problem, we see that the convergence behaviour for the case that is based on the steepest descent method for the ω_j in the determination of the solution for $f - 1$ is much better than the results that correspond to the minimisation of $\mathbf{r}_i$ with respect to ω_j.

Both choices for the Chebyshev nodes do not lead to an improvement in the convergence behaviour of IDR(s) with $s = 8$ and we do not expect different results for other values of s. We conclude that also for the car problem, this approach does not lead to better computational results.

Chapter 8

Conclusions

Enough research will tend to support your conclusions.
Arthur Bloch

Very much hard work remains to be done and one needs not only great perspicacity but often a degree of good fortune.
Christiaan Huygens

In this thesis we considered the acoustics of a car and we studied several techniques to solve the numerical linear systems of equations that result from the discretisation (see chapter 3) of the equations that describe the pressure perturbations in the air inside the car (see chapter 2) and of the displacement of the car structure parts. Since for every frequency the pressure perturbations evolve differently, we solved the linear systems for sequences of frequencies with the aid of various numerical methods known as Krylov subspace methods (see chapter 4). These methods allow the use of results that are already obtained and the research done considers the use of this information for improving the performance of the numerical methods.

We focussed on IDR(s) (see chapter 6), a Krylov subspace method that performs very well on difficult problems, and considered as preconditioner the shifted Laplace preconditioner (see chapter 5). The preconditioner is not needed for just speeding up to convergence, but also for enabling iterative methods to determine the solution. Therefore, the (incomplete) LU factorisation of this preconditioner should not lead to factors that are too crude an approximation to this preconditioner.

In the experiments (see chapter 7) we consider several ways to use the available information. First, we focused on solution vectors of nearby frequencies and applied them to improve the initial guess for the solution vector and to span an initial search space. Second, we considered results on the number of iterations, which allows us to update the shifted Laplace preconditioner in a proper way. Third, we investigated if using approximate eigenvalues of the system matrices reduces the number of iterations.

8.1 Summary of results

We firstly focussed on the fluid part and the structure part of the problem separately, after which we considered the complete car problem. We have seen that the fluid problem needs far less iterations than the structure problem, which in turn is easier to solve than the car problem. All problems become much more difficult for higher frequencies f, since the system matrices get increasingly ill-conditioned when f grows. Extrapolation of earlier obtained solutions to certain frequencies for the initial guess of the upcoming frequency reduces the number of iterations for all problems, and spanning an initial search space with these solutions leads to even better computational results for the fluid and structure problem. Applying both search space and extrapolation for these problems does not significantly effect the convergence compared to the case where we

only use the initial search space. For the car problem, there is hardly any difference in the computational results between using the initial search space or not and we therefore consider only extrapolation of solutions for this problem.

We considered three types of shifted Laplace preconditioners, which all change the system into a form that is more suitable for the numerical methods. The first preconditioner has a purely imaginary shift, the second has a real shift and the third equals the real part of the second preconditioner, which means that the complete damping term and parts of the stiffness matrix are discarded. This latter preconditioner is not used for the fluid problem since it is the same as the real shifted preconditioner for this problem. If we choose the real shifted Laplace preconditioner with a shift close to the frequency of the system, very few iterations are needed for convergence. We therefore update the shift of this preconditioner if too many iterations are needed for a certain frequency and we have seen that this more than halves the computation time for the fluid and structure problem. Fine-tuning of the moment of updating and the precise value of the new shift leads to an additional reduction of 30% (for the fluid problem) or 20% (for the structure structure). For the complete car problem, it appeared that a similar updating strategy for the modified shifted Laplace preconditioner leads to even better results in terms of computation time.

The performance of IDR(s) on all problems is highly dependent on the value of s. For too small values of s, the number of iterations that are needed for convergence for problems with higher frequencies becomes very large, while too high values of s can result in an ill-conditioned matrix in IDR(s). The best results are obtained for $s = 8$ and this choice does not suffer from the above two problems. Compared to some other Krylov subspace methods, IDR(8) performs also very well. It outperforms CGS and BiCGSTAB, of which the convergence behaviour for successive frequencies is very irregular, and GMRES, which needs very few numbers of iterations that smoothly increase for growing frequencies, but also requires long recurrences.

During the iterates of IDR(s) we can obtain a set of approximations to the eigenvalues, the so-called Ritz values. We used these values to determine a subset in the complex plane that hopefully encloses the spectrum and a polynomial that is small on this set. We matched the roots of this polynomial with the roots of the part of the residual polynomial of IDR(s) that depends on ω_j, but this did not lead to better convergence behaviour for our problems.

We have seen that the best approach to this car problem is to solve it using IDR(8), where we use (the exact LU factors of) the modified shifted Laplace preconditioner with proper updates of the shift and extrapolation of previously obtained solution vectors for an initial guess.

8.2 Future research

We list a few suggestions for future research. First, we consider ways to improve the performance of IDR(s).

- The vectors that span the initial search space or the corresponding vectors in $\mathcal{G}_0$ should maybe be adapted in IDR(s) in order to make it possible that the number of iterations is less than $s + 1$.
- For the car problem, the matrix $\mathbf{M}$ in IDR(s) is ill-conditioned. This causes a build-up of rounding errors and hence less accurate or even inaccurate solution vectors. Factorisation of this matrix, with for instance the singular value decomposition, might lead to an improvement in the robustness of IDR(s).

- We used IDR(s) as described in section 6.2, which is based on a so-called orthogonal residual approach. As an alternative, we could instead use QMRIDR(s), which is discussed in Appendix A and relies on a minimal residual approach. Algorithms based on this last approach leads in general to more stable updates for systems that cause breakdown for algorithms with an approach of orthogonal residuals.
- The dissimilarity in Ritz values that follow from different choices for ω_j suggest that the eigenvalue approximations are dependent on ω_j, even though the analysis imply that the Hessenberg matrix is independent from ω_j, see (6.11). In addition, there is a lack of accuracy, which is presumably caused by numerical instability. Research on these matters might result in changes in the current straightforward implementation of the equations of section 6.4 and lead to better eigenvalue approximations.
- Based on the convergence process per Ritz value, it might be possible to determine if a certain Ritz value converges to an eigenvalue or not. If it does not converge, it is clearly not an approximate eigenvalue, while if it converges to a value that is not in the spectrum, it will be interesting to determine what significance this value has.

Some other matters that may also require attention are listed below.

- We considered for ω_j the reciprocals of a few Chebyshev nodes that follows from the Chebyshev polynomial on an ellipse. In addition to this specific polynomial on this particular set, other options are available. We could for instance consider a Chebyshev polynomial on several disjoint sets. It is also possible to use another polynomial, for instance a polynomial based on a Leja sequency. This polynomial $L_i^{\mathcal{C}}(\xi)$ of degree i consists of Leja points $\xi_i \in \mathcal{C}$ that are determined by iteratively chosing them so that the Leja polynomial $L_{i-1}^{\mathcal{C}}(\xi) = \prod_{k=1}^{i-1} |\xi - \xi_k|$ is maximised on $\mathcal{C}$ for $\xi = \xi_i$.
- We used Matlab for all implementations and made a start with the implementation in Fortran. With the use of advanced Fortran packages on numerical linear algebra, it is be possible to further reduce the computation time. Especially parallelisation of the computations will result in a significant reduction.

Appendix A

Quasi Minimal Residual Induced Dimension Reduction (s)

Please, sir, I want some more.
Oliver in *Oliver Twist* (Charles Dickens).

The QMRIDR variant of the IDR algorithm is published in [8]. The idea behind this variant is to apply the GMRES philosophy. As we have seen in section 4.3, GMRES generates a set of normal basis vectors that span the Krylov space, after which a much smaller and projected system can be solved. Here, a set of vectors $\mathbf{g}_i \in \mathcal{G}_j$ is orthonormalised by using Arnoldi iteration (see section 4.2) and this set is used to rewrite the original system into a smaller system.

Given the unit vector $\mathbf{g}_1 = \mathbf{r}_0/\|\mathbf{r}_0\| \in \mathcal{G}_0$, we compute equivalent to (4.3)

$$\mathbf{g}_{i+1} = \frac{\mathbf{A}\mathbf{g}_i - \sum_{k=1}^{i} \beta_{k,i}\mathbf{g}_k}{\beta_{i+1,i}} \quad (i \leq s),$$

with $\beta_{k,i} = \mathbf{A}\mathbf{g}_k \cdot \mathbf{g}_i$ and $\beta_{i+1,i} = \|\mathbf{A}\mathbf{g}_i - \sum_{k=1}^{i} \beta_{k,i}\mathbf{g}_k\|$, and combine these equations to obtain

$$\mathbf{G}_{s+1}\underline{\mathbf{H}}_s = \mathbf{A}\mathbf{G}_s, \tag{A.1}$$

where $\mathbf{G}_s = (\mathbf{g}_1, \ldots, \mathbf{g}_s)$ and $\mathbf{G}_{s+1} = (\mathbf{G}_s, \mathbf{g}_{s+1})$. For the computation of the first vector $\mathbf{g}_{i+1} \in \mathcal{G}_j$ (for $j = 1, 2, \ldots$) the vectors $\mathbf{g}_{i-s}, \ldots, \mathbf{g}_i$ are available. First we obtain the vector $\mathbf{v}_i$ by substituting (6.5) into (6.3) and then we compute the vector $\mathbf{t}_i = (\mathbf{A} - \mu_j\mathbf{I})\mathbf{v}_i \in \mathcal{G}_j$. We set $\mathbf{g}_{i+1} = \mathbf{t}_i/\|\mathbf{t}_i\|$ and substitution of (6.5) into this equation results in

$$\mathbf{A}\left(\mathbf{g}_i - \sum_{k=i-s}^{i-1} \gamma_k^i \mathbf{g}_k\right) = \mu_j\left(\mathbf{g}_i - \sum_{k=i-s}^{i-1} \gamma_k^i \mathbf{g}_k\right) + \beta_{i+1,i}\mathbf{g}_{i+1}.$$

The vectors $\mathbf{g}_{i+2}, \ldots, \mathbf{g}_{i+s+1}$ are obtained iteratively by computing $\mathbf{t}_{i+1} = (\mathbf{A} - \mu_j\mathbf{I})\mathbf{v}_{i+1}$ and

$$\mathbf{g}_{i+1} = \frac{\mathbf{t}_{i+1} - \sum_{k=j(s+1)+1}^{i} \beta_{k,i}\mathbf{g}_k}{\beta_{i+1,i}}.$$

We remark that all parameters $\beta_{k,i}$ are chosen such that the vectors $\mathbf{g}_i \in \mathcal{G}_j$ are orthonormal to each other. We obtain

$$\mathbf{A}\left(\mathbf{g}_i - \sum_{k=i-s}^{i-1} \gamma_k^i \mathbf{g}_k\right) = \mu_j\left(\mathbf{g}_i - \sum_{k=i-s}^{i-1} \gamma_k^i \mathbf{g}_k\right) + \sum_{k=j(s+1)+1}^{i+1} \beta_{k,i}\mathbf{g}_k,$$

and with a proper choice of the matrices $\mathbf{U}_i$ and $\underline{\mathbf{H}}_i$ this results in the generalised Hessenberg decomposition

$$\mathbf{A}\mathbf{G}_i\mathbf{U}_i = \mathbf{G}_{i+1}\underline{\mathbf{H}}_i. \tag{A.2}$$

Note that for the first s $\mathbf{g}$-vectors we can write (A.1) in this form by defining $\mathbf{U}_s = \mathbf{I}$. The matrix $\mathbf{U}_i$ is an $i \times i$ upper triangular matrix with upper bandwidth s and $\underline{\mathbf{H}}_i$ is an extended Hessenberg matrix with upper bandwidth s and size $(i+1) \times i$.

The Quasi Minimal Residual (QMR) method [6] can be used to solve the linear system (A.2) and its derivation is similar to the derivation of GMRES for solving (4.1) (see §4.4). We use (A.2) to construct approximation vectors $\mathbf{x}_i$ that are a linear combination of $\mathbf{g}_{i-s+1}, \ldots, \mathbf{g}_i$ by

$$\mathbf{x}_i = \mathbf{x}_0 + \mathbf{G}_i\mathbf{U}_i\mathbf{y}_i,$$

where $\mathbf{y}_i$ is obtained by solving the minimisation problem

$$\min_{\mathbf{y}_i \in \mathbb{C}^i} \|\mathbf{G}_{i+1}(\beta\mathbf{e}_1 - \underline{\mathbf{H}}_i\mathbf{y}_i)\|,$$

and $\beta = \|\mathbf{r}_0\|$. The matrix $\mathbf{G}_{i+1}$ does not consist of orthonormal vectors and we therefore minimise an upper bound. That is, we solve

$$\min_{\mathbf{y}_i \in \mathbb{C}^i} \|\mathbf{G}_{i+1}\| \, \|\beta\mathbf{e}_1 - \underline{\mathbf{H}}_i\mathbf{y}_i\|. \tag{A.3}$$

It can be shown that $\|\mathbf{G}_{i+1}\| \leq \sqrt{\lceil (i+1)/(s+1) \rceil}$ [8].

We obtain the solution to (A.3) by the QR decomposition of $\underline{\mathbf{H}}_i$ as $\widetilde{\mathbf{Q}}_{i+1}\underline{\mathbf{H}}_i = \underline{\mathbf{R}}_i$. This is done by left multiplication of $\underline{\mathbf{H}}_i$ with a sequence of Givens matrices, similar to §4.4. The solution of (A.3) and the approximation $\mathbf{x}_i$ are then given by

$$\mathbf{y}_i = \beta\underline{\mathbf{R}}_i^{-1}\widetilde{\mathbf{Q}}_{i+1}^*\mathbf{e}_1 \quad \text{and} \quad \mathbf{x}_i = \mathbf{x}_0 + \beta\mathbf{G}_i\mathbf{U}_i\underline{\mathbf{R}}_i^{-1}\widetilde{\mathbf{Q}}_{i+1}^*\mathbf{e}_1.$$

To avoid the long recurrence problem of increasing size of the matrix $\mathbf{G}_i$, we compute $\mathbf{x}_i$ without the need of this matrix $\mathbf{G}_i$. We define the matrices $\mathbf{V}_i = (\mathbf{v}_1, \ldots, \mathbf{v}_i) = \mathbf{G}_i\mathbf{U}_i$ of which the column vectors satisfy (6.5), and $\mathbf{W}_i = (\mathbf{w}_1, \ldots, \mathbf{w}_i) = \mathbf{V}_i\widetilde{\mathbf{R}}_i^{-1}$ and it follows that

$$\mathbf{w}_i = \frac{1}{r_{i_{i,i}}}\left(\sum_{k=i-s-1}^{i-1} \mathbf{w}_k r_{i_{k,i}}\right).$$

After defining the vector $\boldsymbol{\phi}_i = \beta\widetilde{\mathbf{Q}}_{i+1}^*\mathbf{e}_1$, we obtain the short recurrence update

$$\mathbf{x}_i = \mathbf{x}_{i-1} + \mathbf{w}_i\phi_{i_i}.$$

The above can be generalised for solving multi-shift systems.

Multi-shift QMRIDR(s)

For many Krylov space methods a shifted version is developed. We refer to [12] for a discussion of the principles of constructing these methods. The idea is to solve shifted versions of the linear system (4.1),

$$(\mathbf{A} - \sigma\mathbf{I})\mathbf{x}^\sigma = \mathbf{b},$$

for a whole set of values of σ simultaneously. The above calculations can be used to derive a multi shift version of QMRIDR(s). We note that the direction of the first residuals need to be

independent of σ hence we need $\mathbf{g}_1 = \mathbf{r}_0^\sigma / \|\mathbf{r}_0^\sigma\|$ for all shifted systems. It is therefore reasonable to set $\mathbf{x}_0^\sigma = \mathbf{0}$ and hence $\mathbf{r}_0^\sigma = \mathbf{b}$.

We construct the approximate solutions $\mathbf{x}_i^\sigma$ similar to the unshifted system by

$$\mathbf{x}_i^\sigma = \mathbf{G}_{i+1} \underline{\mathbf{U}}_i \mathbf{y}_i^\sigma,$$

where $\underline{\mathbf{U}}_i = (\mathbf{U}_i^T, \mathbf{0})^T$ and obtain $\mathbf{y}_i^\sigma$ by solving the minimisation problem

$$\min_{\mathbf{y}_i^\sigma \in^i} \|\mathbf{G}_{i+1}\| \, \|\beta \mathbf{e}_1 - [\underline{\mathbf{H}}_i - \sigma \underline{\mathbf{U}}_i] \mathbf{y}_i^\sigma\|,$$

which is an upper bound for

$$\min_{\mathbf{y}_i^\sigma \in^i} \|\mathbf{G}_{i+1}(\beta \mathbf{e}_1 - [\underline{\mathbf{H}}_i - \sigma \underline{\mathbf{U}}_i] \mathbf{y}_i^\sigma)\|.$$

We compute QR decompositions of $\underline{\mathbf{H}}_i - \sigma \underline{\mathbf{U}}_i$ to obtain matrices $\widetilde{\mathbf{Q}}_{i+1}^\sigma$ and $\underline{\mathbf{R}}_i^\sigma$ that satisfy $\widetilde{\mathbf{Q}}_{i+1}^\sigma (\underline{\mathbf{H}}_i - \sigma \underline{\mathbf{U}}_i) = \underline{\mathbf{R}}_i^\sigma$. We define the vectors $\boldsymbol{\phi}_i^\sigma$ and update vectors $\mathbf{w}_i^\sigma$ analogous to the unshifted case, after which the approximations $\mathbf{x}_i^\sigma$ can be computed by

$$\mathbf{x}_i^\sigma = \mathbf{x}_{i-1}^\sigma + \mathbf{w}_i^\sigma \phi_{i_i}^\sigma.$$

We present the multi-shift QMRIDR(s) algorithm below.

Algorithm 5 Multi-shift Quasi Minimal Residual Induced Dimension Reduction (s).

Require: $\sigma_i \in \mathbb{C}$, $(i = 1, \cdots, n_\sigma)$, $s > 0$, $\mathbf{P}_0$, tol $\in (0, 1)$, maxit > 0
  $\mathbf{x}^{\sigma_i} = \mathbf{0}$, $(i = 1, \cdots, n_\sigma)$, $\mu = 0$, $\mathbf{M} = \mathbf{0}$, $\mathbf{G} = \mathbf{0}$, $\mathbf{w} = \mathbf{0}$, $\kappa = 0.7$
  $\mathbf{g} = \mathbf{b}$ $\rho = \|\mathbf{g}\|$, $\rho_0 = \rho$; $\mathbf{g} = \frac{1}{\rho}\mathbf{g}$
  for $i = 1, \ldots, n_\sigma$ **do**
    $\mathbf{W}^{\sigma_i} = \mathbf{0}$, $\mathbf{cs}^{\sigma_i} = \mathbf{0}$, $\mathbf{sn}^{\sigma_i} = \mathbf{0}$, $\phi^{\sigma_i} = 0$, $\hat{\phi}^{\sigma_i} = \|\mathbf{g}\|$
  end for
  $n = 0$, $j = 0$
  while $\rho/\rho_0 >$ tol and $n \leq$ maxit **do**
    for $k = 1, \ldots, s+1$ **do**
      $n = n + 1$
      $\underline{\mathbf{u}} = \mathbf{0}, \underline{\mathbf{u}}_{(s+1)} = 1$
      $\mathbf{m} = \mathbf{P}_0^* \mathbf{g}$
      if $n > s$ **then**
        Solve $\boldsymbol{\gamma}$ from $\mathbf{M}\boldsymbol{\gamma} = \mathbf{m}$
        $\mathbf{v} = \mathbf{g} - \mathbf{G}\boldsymbol{\gamma}$
        $\underline{\mathbf{u}}_{(1:s)} = -\boldsymbol{\gamma}$
      else
        $\mathbf{v} = \mathbf{g}$
      end if
      $\mathbf{M}_{(:,1:s-1)} = \mathbf{M}_{(:,2:s)}$, $\mathbf{M}_{(:,s)} = \mathbf{m}$
      $\mathbf{G}_{(:,1:s-1)} = \mathbf{G}_{(:,2:s)}$, $\mathbf{G}_{(:,s)} = \mathbf{g}$
      $\mathbf{g} = \mathbf{A}\mathbf{v}$;
      if $k = s + 1$ **then**
        $j = j + 1$
        $\omega = (\mathbf{g} \cdot \mathbf{v})/(\mathbf{g} \cdot \mathbf{g})$

$\rho = (\mathbf{g} \cdot \mathbf{v})/(\|\mathbf{g}\| \, \|\mathbf{v}\|)$
if $|\rho| < \kappa$ **then**
$\omega = \omega\kappa/|\rho|$
end if
if $|\omega| >$ `eps` **then**
$\mu = 1/\omega$
else
$\mu = \sqrt{\|\mathbf{A}\|_1 \|\mathbf{A}\|_\infty}$
end if
end if
$\mathbf{g} = \mathbf{g} - \mu\mathbf{v}$
$\underline{\mathbf{h}} = \mu\underline{\mathbf{u}}$
if $k < s + 1$ **then**
$\beta = \mathbf{G}^*_{(:,s-k+1:s)}\mathbf{g}$, $\mathbf{g} = \mathbf{g} - \mathbf{G}_{(:,s-k+1:s)}\beta$
$\hat{\beta} = \mathbf{G}^*_{(:,s-k+1:s)}\mathbf{g}$, $\mathbf{g} = \mathbf{g} - \mathbf{G}_{(:,s-k+1:s)}\hat{\beta}$
$\beta = \beta + \hat{\beta}$
$\underline{\mathbf{h}}_{(s+1-k+1:s+1)} = \underline{\mathbf{h}}_{(s+1-k+1:s+1)} + \beta$
end if
$\underline{\mathbf{h}}_{(s+2)} = \|\mathbf{g}\|$, $\mathbf{g} = \mathbf{g}/\|\mathbf{g}\|$
$\rho = 0$
for $i = 1, \ldots, n_\sigma$ **do**
$\mathbf{r} = \mathbf{0}$, $\mathbf{r}_{(2:s+3)} = \underline{\mathbf{h}} - \sigma_i\underline{\mathbf{u}}$
$l_b = \max(1, s + 3 - n)$
for $l = l_b, \ldots, s + 1$ **do**
$t = \mathbf{r}_{(l)}$
$\mathbf{r}_{(l)} = \mathbf{cs}^{\sigma_i}_{(l)} t + \mathbf{sn}^{\sigma_i}_{(l)}\mathbf{r}_{(l+1)}$
$\mathbf{r}_{(l+1)} = -\overline{\mathbf{sn}}^{\sigma_i}_{(l)} t + \mathbf{cs}^{\sigma_i}_{(l)}\mathbf{r}_{(l+1)}$
end for
if $|\mathbf{r}_{(s+2)}| <$ `eps` **then**
$\mathbf{cs}^{\sigma_i}_{(s+2)} = 0$, $\mathbf{sn}^{\sigma_i}_{(s+2)} = 1$, $\mathbf{r}_{(s+2)} = \mathbf{r}_{(s+3)}$
else
$t = |\mathbf{r}_{(s+2)}| + |\mathbf{r}_{(s+3)}|$
$\rho = t\sqrt{(\mathbf{r}_{(s+2)}/t)^2 + (\mathbf{r}_{(s+3)}/t)^2}$
$\alpha = \mathbf{r}_{(s+2)}/|\mathbf{r}_{(s+2)}|$
$\mathbf{cs}^{\sigma_i}_{(s+2)} = |\mathbf{r}_{(s+2)}|/\rho$, $\mathbf{sn}^{\sigma_i}_{(s+2)} = \alpha\overline{\mathbf{r}_{(s+3)}}/\rho$, $\mathbf{r}_{(s+2)} = \alpha\rho$
end if
$\phi^{\sigma_i} = \mathbf{cs}^{\sigma_i}_{(s+2)}\hat{\phi}^{\sigma_i}$, $\hat{\phi}^{\sigma_i} = -\overline{\mathbf{sn}}^{\sigma_i}_{(s+2)}\hat{\phi}^{\sigma_i}$
$\mathbf{cs}^{\sigma_i}_{(:,1:s+1)} = \mathbf{cs}^{\sigma_i}_{(:,2:s+2)}$, $\mathbf{sn}^{\sigma_i}_{(:,1:s+1)} = \mathbf{sn}^{\sigma_i}_{(:,2:s+2)}$
$\mathbf{w} = (\mathbf{v} - \mathbf{W}^{\sigma_i}\mathbf{r}_{(1:s+1)})/\mathbf{r}_{(s+2)}$
$\mathbf{W}^{\sigma_i}_{(:,1:s)} = \mathbf{W}^{\sigma_i}_{(:,2:s+1)}$, $\mathbf{W}^{\sigma_i}_{(:,s+1)} = \mathbf{w}$
$\mathbf{x}^{\sigma_i} = \mathbf{x}^{\sigma_i} + \phi^{\sigma_i}\mathbf{w}$
$\rho = \max(\rho, |\hat{\phi}|\sqrt{j+1})$
end for
end for
end while

Appendix B

Initial experiments

Computers are useless. They can only give you answers.
Pablo Ruiz y Picasso

We investigate the properties of the shifted Laplace preconditioner

$$\mathbf{P}^i(k_0) = \mathbf{K} + ik_0\mathbf{C} + ik_0^2\mathbf{M},$$

and the preconditioner that is optimal for a single frequency

$$\mathbf{P}^r(k_0) = \mathbf{K} + ik_0\mathbf{C} - k_0^2\mathbf{M},$$

where k_0 is held fixed and chosen such that $z = -ik_0^2$ with z as defined in (5.6). Both $\mathbf{P}^i(k_0)$ and $\mathbf{P}^r(k_0)$ are applied to the linear system that describes the acoustics in a 4×4 m^2 room with a point source in the centre. The wall at the east is absorbing with an acoustic impedance $Z_n = \frac{1}{5} - \frac{3}{2}i$ kg/(s m^2) and the other walls are reflecting (see section 3.1). We choose the speed of sound in air c_0 is equal to 340 m/s. So, we consider the system (4.1) where the system matrix is given by

$$\mathbf{A}(k_i) = \mathbf{K} + ik_i\mathbf{C} - k_i^2\mathbf{M},$$

for a sequence of acoustic wave numbers $k_i = k_1, \ldots, k_n \in \mathbb{R}^+$.

In the first set of experiments we consider the location of the eigenvalues of the system matrix of the linear system (4.1) and the equivalent preconditioned system (4.9) for a single frequency k and different values of k_0. In the second set we study the computation time of the (incomplete) LU factorisation of the preconditioners $\mathbf{P}^i(k_0)$ and $\mathbf{P}^r(k_0)$. In the third experiments we apply several Krylov space methods to solve the linear systems (4.1) and (4.9) for a range of frequencies and fixed k_0. In the last experiments we apply IDR(4) to (4.9) where we vary k_0.

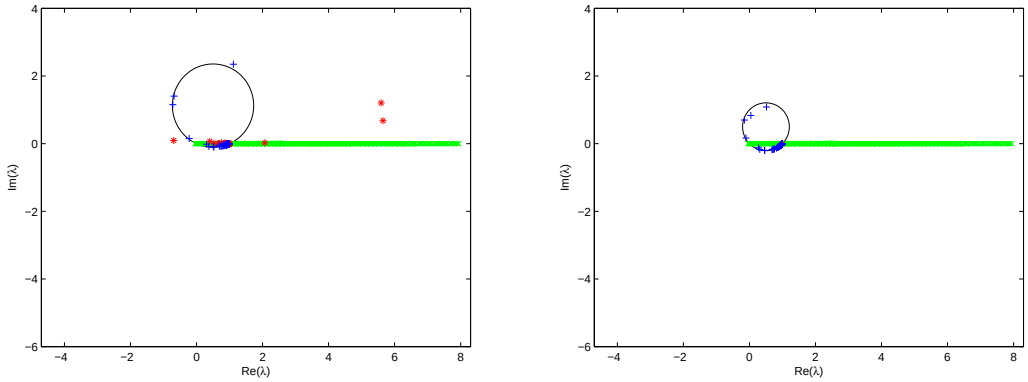
All experiments are performed on a PC with a single CPU clocked at 2.1GHz and with 8Gb RAM with Matlab 7.7.

B.1 Location of the eigenvalues

For the problem of the two-dimensional room we choose a gridsize $h = L/25 = 16$ cm and investigate the location of the eigenvalues of the matrices $\mathbf{A}$, $[\mathbf{P}^i(k_0)]^{-1}\mathbf{A}$ and $[\mathbf{P}^r(k_0)]^{-1}\mathbf{A}$, all of size 676×676. We consider a frequency $f = 60$ Hz (and hence $k = 60 \cdot 2\pi/c_0 \approx 1.1088$ m^{-1}) and compare the preconditioners $\mathbf{P}^i(k_0)$ and $\mathbf{P}^r(k_0)$ for values of k_0 that correspond to the frequencies 40, 60, 80 and 100 Hz. In this way, we investigate if it is reasonable to choose the preconditioner with fixed k_0 for a range of frequencies.

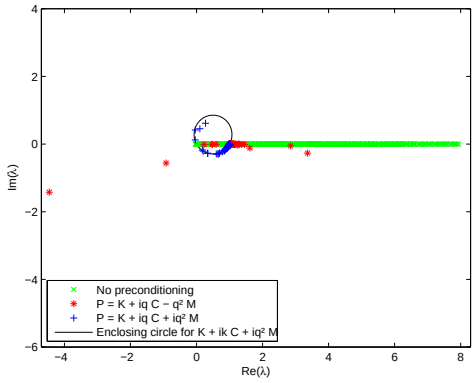
The location of the eigenvalues is displayed in Figure B.1. For $k_0 = k$ the eigenvalues of $(\mathbf{P}^i)^{-1}\mathbf{A}$ are on or inside the circle (which is defined by (5.4)) and the eigenvalues of $(\mathbf{P}^r)^{-1}\mathbf{A} = \mathbf{I}$ are of

course equal to 1, see Figure B.1(b). For $k_0 \neq k$ we see from Figures B.1(a), (c) and (d) that the eigenvalues of $(\mathbf{P}^i)^{-1}\mathbf{A}$ are more clustered than the eigenvalues of $(\mathbf{P}^r)^{-1}\mathbf{A}$. In addition, the eigenvalues of $(\mathbf{P}^i)^{-1}\mathbf{A}$ are not too far from the circle (which enclosed the eigenvalues of $(\mathbf{K} + ik\mathbf{C} + iq^2\mathbf{M})^{-1}\mathbf{A}$) while the distribution of the eigenvalues of $(\mathbf{P}^r)^{-1}\mathbf{A}$ varies strongly, even for small differences in values of k_0.

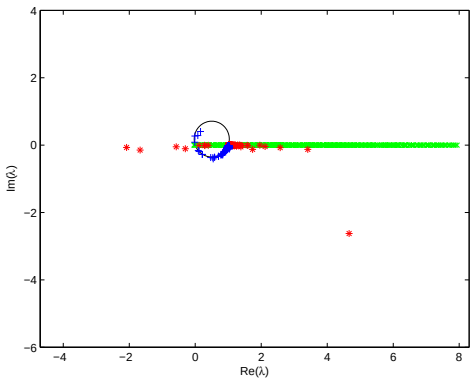


(a) $k = 60 \cdot 2\pi/c_0$, $k_0 = 40 \cdot 2\pi/c_0$

(b) $k = 60 \cdot 2\pi/c_0$, $k_0 = 60 \cdot 2\pi/c_0$



(c) $k = 60 \cdot 2\pi/c_0$, $k_0 = 80 \cdot 2\pi/c_0$



(d) $k = 60 \cdot 2\pi/c_0$, $k_0 = 100 \cdot 2\pi/c_0$

Figure B.1: Location of the eigenvalues.

The convergence of Krylov space methods is generally better for problems where the system matrix is not highly indefinite and the eigenvalues are clustered away from zero. For the cases with $k_0 \neq k$, the eigenvalues of both $(\mathbf{P}^i)^{-1}\mathbf{A}$ and $(\mathbf{P}^r)^{-1}\mathbf{A}$ are mainly clustered around 1 and some eigenvalues are close to 0. A disadvantage of $(\mathbf{P}^r)^{-1}\mathbf{A}$ is that this matrix turns out to be highly indefinite.

The problem of 676 equations is very small, but the location of the eigenvalues of much larger matrices appears to be very similar to the respective small matrices.

B.2 LU and ILU factorisation

Since we need to apply the inverse of the preconditioners $\mathbf{P}^i(k_0)$ and $\mathbf{P}^r(k_0)$, we decompose these preconditioners into an upper and lower triangular matrix using (incomplete) LU factorisation. The exact LU factorisation of for instance $\mathbf{P}^i$ is very time-consuming since it is large and it results in matrices $\mathbf{L}^i$ and $\mathbf{U}^i$ such that $\mathbf{L}^i\mathbf{U}^i = \mathbf{P}^i$. Incomplete factorisation results in approximations $\widetilde{\mathbf{L}}^i$ and $\widetilde{\mathbf{U}}^i$ with $\mathbf{P}^i \neq \widetilde{\mathbf{L}}^i\widetilde{\mathbf{U}}^i$ and it is possible that the properties of the original preconditioner $\mathbf{P}^i$ are lost.

We use three types of LU factorisation, namely the standard (UMFPACK) LU factorisation, the ILU(0) factorisation, which allows no fill-in, and the ILUT factorisation with a drop tolerance $\tau = 0.01$. This drop tolerance is applied in such a way that the non-zero off-diagonal entries of $\widetilde{\mathbf{U}}^i$ satisfy $|u_{j,k}| \geq \tau\|\mathbf{P}^i_{(:,k)}\|$ and for the non-zeros of $\widetilde{\mathbf{L}}^i$ we have that $|l_{j,k}| \geq \tau\|\mathbf{P}^i_{(:,k)}\|/u_{k,k}$. All these methods are available in Matlab.

	LU			ILU(0)			ILUT(0.01)		
gridsize h	$L/99$	$L/199$	$L/399$	$L/399$	$L/799$	$L/1599$	$L/199$	$L/399$	$L/799$
$n \times 1000$	10	40	160	160	640	1280	40	160	640
$\mathbf{P}^i(k_0)$	1.74	27.9	420	0.06	0.27	1.26	2.03	21.2	177
$\mathbf{P}^r(k_0)$	1.79	26.3	406	0.06	0.22	1.10	1.50	17.1	136

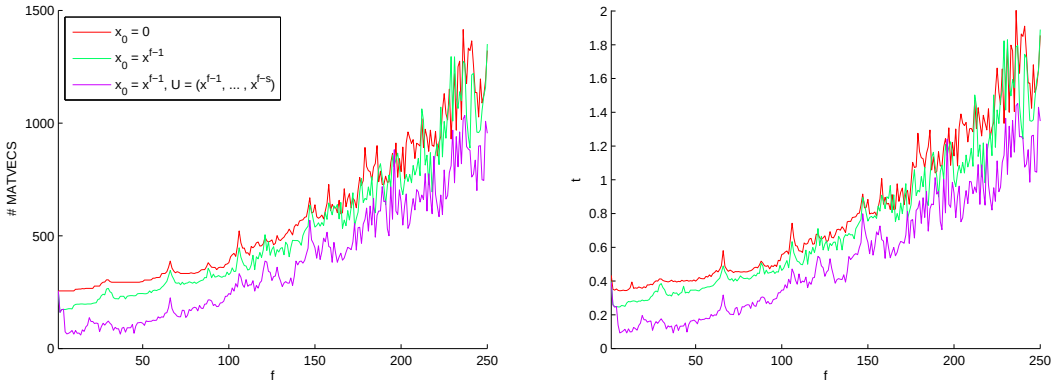
Table B.1: Computation time (s) of (I)LU factorisation, $k_0 = 125 \cdot 2\pi/c_0$.

We study the time it takes to apply the above (incomplete) LU factorisations. Some results are tabulated in Table B.1, where n equals the number of equations. We see that LU factorisation is very time consuming and of order $\mathcal{O}(h^4)$, while ILU(0) needs far less computation time and is of order $\mathcal{O}(h^2)$. The order of ILUT(τ) is highly dependent on the value of τ and we see that ILUT(0.01) is roughly of order $\mathcal{O}(h^3)$. This means that if we extrapolate to one million equations, we need for the LU factorisation of $\mathbf{P}^i$ approximately $16500\,s$, for the ILU(0) factorisation $0.77\,s$ and for the ILUT(0.01) factorisation $346\,s$.

B.3 Krylov subspace methods

We apply (I)LU factorisation to the preconditioners $\mathbf{P}^i(k_0)$ and $\mathbf{P}^r(k_0)$ with $k_0 = 125 \cdot 2\pi/c_0$ and compute the solution to (4.1) and (4.9) for a range of frequencies $f = 1, \ldots, 250$ Hz with the Krylov subspace methods IDR(4), IDR(8), CGS, BiCGSTAB, QMR and GMRES.

We set tol $= \|\mathbf{b}\| \cdot 10^{-8}$ and maxit $= 2000$. As initial guess we choose for $f = 1$ the zero vector, so $\mathbf{x}_0 = \mathbf{0}$, and for $f \neq 1$ the obtained solution to the previous frequency, that is $\mathbf{x}_0 = \mathbf{x}^{f-1}$. For the computation of frequencies $f > s$ with IDR(s) we use as initial space the space spanned by the last s solution vectors and hence set $\mathbf{U}_0 = (\mathbf{x}^{f-1}, \ldots, \mathbf{x}^{f-s})$. This choice is motivated by the computation of the pressure disturbances with IDR(4) on a mesh with gridsize $h = L/50 = 8$ cm, of which the computational results are presented in Figure B.2. The reduction of the total number of MATVECS (matrix vector multiplications) and the total computation time by choosing $\mathbf{x}_0 = \mathbf{x}^{f-1}$ combined with $\mathbf{U}_0 = (\mathbf{x}^{f-1}, \ldots, \mathbf{x}^{f-s})$ instead of $\mathbf{x}_0 = \mathbf{0}$ is approximately 40%.
We consider a gridsize of $h = L/50 = 8$ cm (and hence the number of equations n equals 2601) and choose $k_0 = 125 \cdot 2\pi/c_0$. The results concerning the computation time are tabulated in Table B.2,



(a) Number of MATVECS per frequency

(b) Computation time (s) per frequency

Figure B.2: Several choices for $\mathbf{x}_0$ and initial $\mathbf{U}_0$.

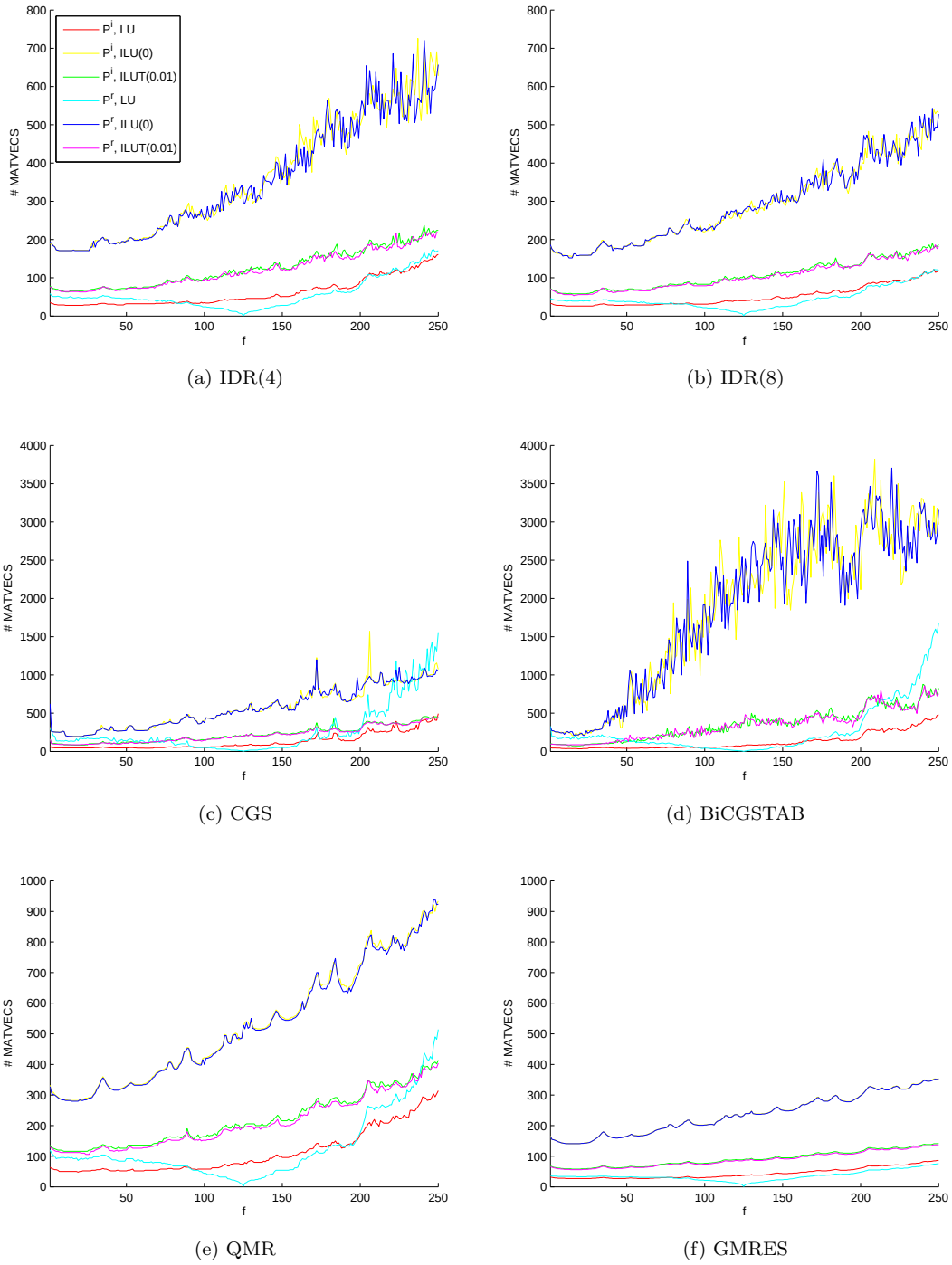
where the computation time for obtaining the (approximate) LU factors of $\mathbf{P}^i$ and $\mathbf{P}^r$ is not taken into account. A $*$ in this table indicates that the method does not converge for all frequencies. In the case of BiCGSTAB, the residuals are almost always still very large after 2000 iterations, while CGS does not converge for a few frequencies and we still have that $\|\mathbf{r}_{2000}\|$ is less than 10 times tol.

For each method, we compare the computation times of the preconditioned system (4.9) with the unpreconditioned system (4.1). If we use the exact LU factorisation, most methods perform slightly better if we use $\mathbf{P}^i$ as preconditioner, but $\mathbf{P}^r$ never improves the computation time. For QMR both preconditioners lead to an increase of computation time while for GMRES the preconditioners greatly reduce the computation time. The ILU(0) and the ILUT(0.01) factorisations of the preconditioners result always in a significant reduction of the computation time. It is remarkable that the results for $\mathbf{P}^i$ and $\mathbf{P}^r$ are close together. If we compare the (I)LU factorisations with each other, we see that both the ILU(0) and the ILUT(0.01) factorisation leads to a further reduction of the computation time than the LU factorisation, and ILUT(0.01) seems to be approximately 1.5 times better in reducing the computation time, compared to ILU(0).

Preconditioner	Factorisation	IDR(4)	IDR(8)	CGS	BiCGSTAB	QMR	GMRES
$\mathbf{P}^i(k_0)$	LU	99	96	215	240	336	157
	ILU(0)	62	69	115	274	133	455
	ILUT(0.01)	37	41	75	94	96	146
$\mathbf{P}^r(k_0)$	LU	256	203	*827	1009	855	208
	ILU(0)	58	69	91	271	124	444
	ILUT(0.01)	31	37	77	118	91	123
No preconditioning		142	135	*233	*787	205	4428

Table B.2: Computation time (s) for each method, $h = L/50$, $k_0 = 125 \cdot 2\pi/c_0$.

Now, we consider a gridsize of $h = L/100 = 4$ cm (so $n = 10201$) and again choose $k_0 = 125 \cdot 2\pi/c_0$. In Table B.3 the computation time is displayed for different methods, where the computation time



(a) IDR(4)

(b) IDR(8)

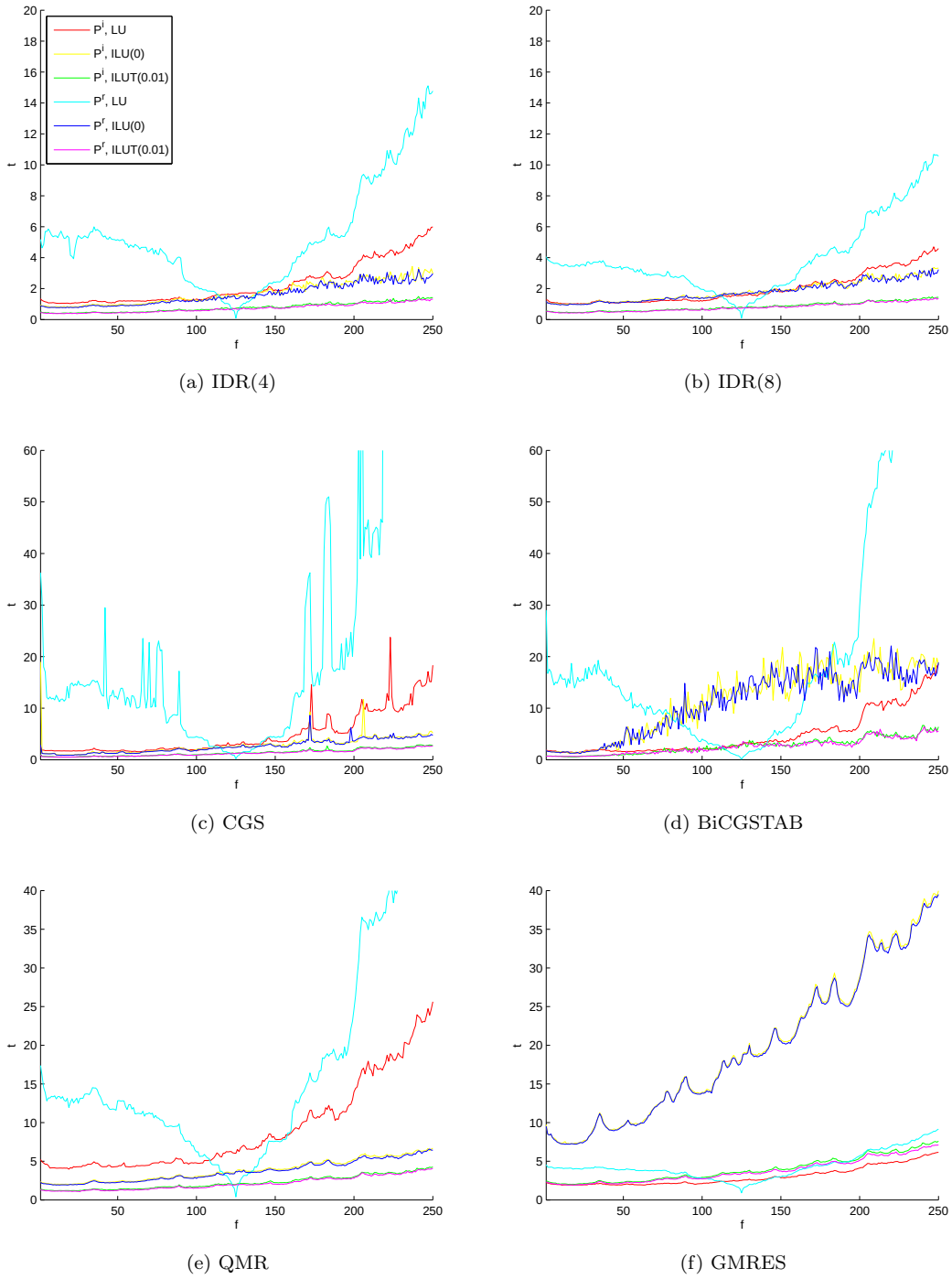
(c) CGS

(d) BiCGSTAB

(e) QMR

(f) GMRES

Figure B.3: Number of MATVECS per frequency for $h = L/100$.



(a) IDR(4)

(b) IDR(8)

(c) CGS

(d) BiCGSTAB

(e) QMR

(f) GMRES

Figure B.4: Computation time (s) per frequency for $h = L/100$.

of (incomplete) LU factorisation is not included. In this table, the same behaviour considering the computation time emerges: if we use exact LU factorisation of $\mathbf{P}^i$, preconditioning with $\mathbf{P}^r$ is worse than preconditioning with $\mathbf{P}^i$, while the results for the incomplete factorisations of $\mathbf{P}^i$ and $\mathbf{P}^r$ are very similar.

Preconditioner	Factorisation	IDR(4)	IDR(8)	CGS	BiCGSTAB	QMR	GMRES
$\mathbf{P}^i(k_0)$	LU	559	493	1200	1209	2260	759
	ILU(0)	428	457	701	2879	953	4953
	ILUT(0.01)	195	207	347	671	562	988
$\mathbf{P}^r(k_0)$	LU	1325	991	7228	5893	4359	1985
	ILU(0)	401	445	649	2776	918	4899
	ILUT(0.01)	181	195	323	636	530	935

Table B.3: Computation time (s) for each method, $h = L/100$, $k_0 = 125 \cdot 2\pi/c_0$.

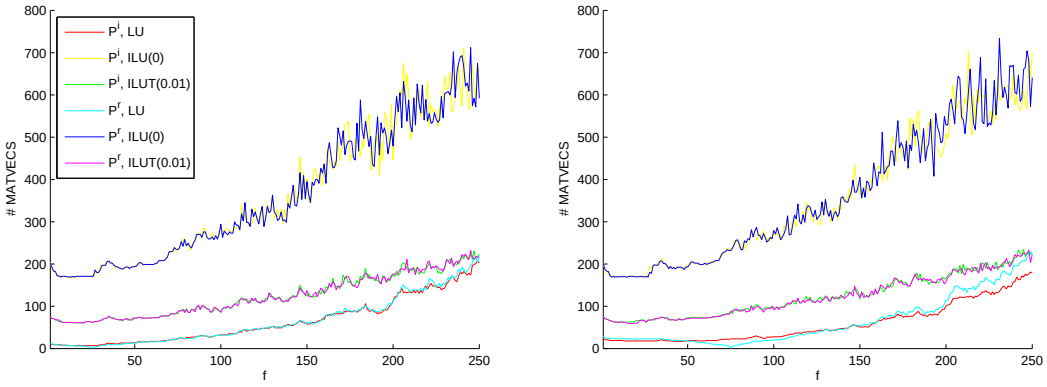
We consider the number of MATVECS and the computation time per frequency and hence investigate if the preconditioners with a fixed k_0 are useful for a range of frequencies where k is close to k_0. The results are given in Figures B.3 and B.4.Since k_0 corresponds to a frequency of 125 Hz, we would expect that on the interval $f \in [100, 150]$ the preconditioners reduce the number of MATVECS and the computation time the most. It is clear that this is indeed the case for $\mathbf{P}^r$ combined with exact LU factorisation, but it seems neither for incomplete LU factorisation nor for any (I)LU factorisation of $\mathbf{P}^i$ the case. We see from these figures also that the difference in preconditioning with $\mathbf{P}^i$ or $\mathbf{P}^r$ combined with incomplete LU factorisation leads to similar behaviour in the number of MATVECS and in computation time.

B.4 IDR(4) with various shifts for the shifted Laplace preconditioner

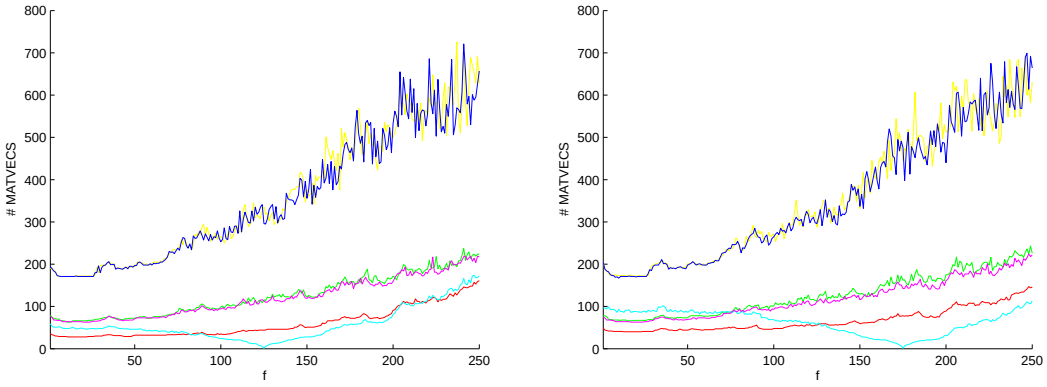
To study the extent to which the convergence properties of $(\mathbf{P}^i(k_0))^{-1}\mathbf{A}$ and $(\mathbf{P}^r(k_0))^{-1}\mathbf{A}$ for IDR(4) depend on the value of k_0, we apply the preconditioners $\mathbf{P}^i(k_0)$ and $\mathbf{P}^r(k_0)$ for different fixed values of k_0. We also consider a set of preconditioners $\mathbf{P}^i(k_0)$ and $\mathbf{P}^r(k_0)$ where $k_0 = f_0 \cdot 2\pi/c_0$ and $f_0 = 25 + 50\lfloor f/50 \rfloor$. The total computation time is given in Table B.4 and the results per frequency are given in Figures B.5 and B.6.

		f_0					
Preconditioner	Factorisation	25	75	125	175	225	$25 + 50\lfloor f/50 \rfloor$
$\mathbf{P}^i(k_0)$	LU	591	547	558	628	748	359
	ILU(0)	432	476	431	431	430	402
	ILUT(0.01)	194	229	196	201	211	177
$\mathbf{P}^r(k_0)$	LU	1379	1370	1408	1528	2551	501
	ILU(0)	410	413	403	404	405	430
	ILUT(0.01)	188	187	184	180	179	203

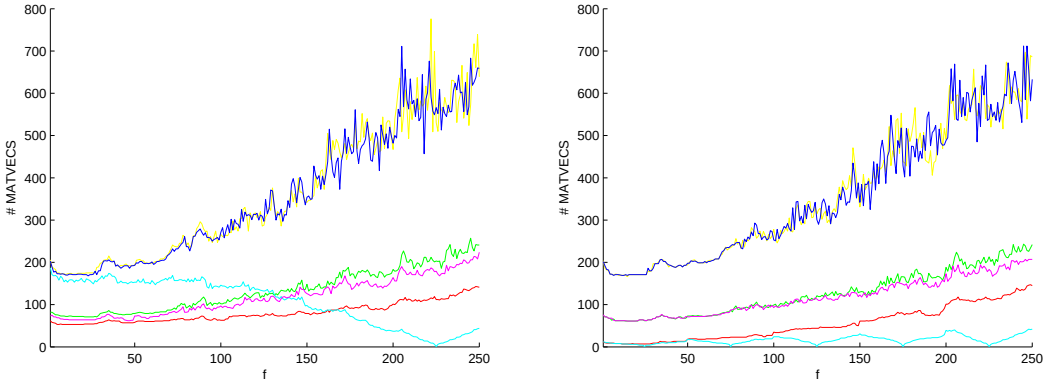
Table B.4: Computation time (s) for IDR(4), $h = L/100$.



(a) $k_0 = 25 \cdot 2\pi/c_0$ (b) $k_0 = 75 \cdot 2\pi/c_0$

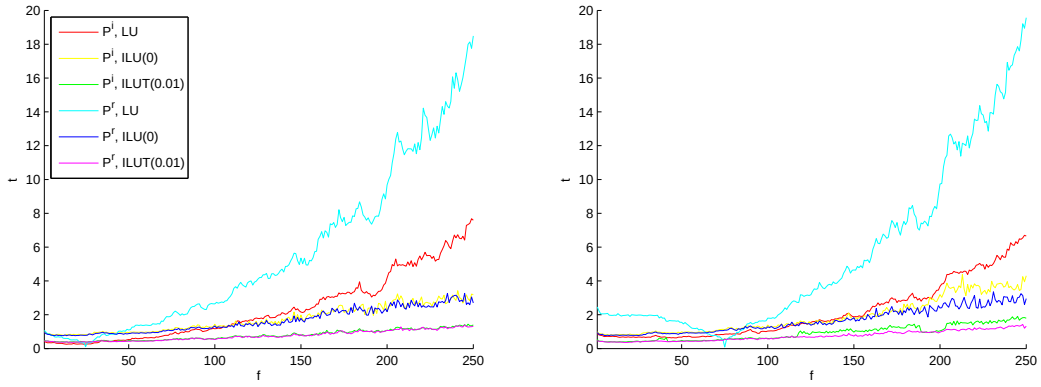


(c) $k_0 = 125 \cdot 2\pi/c_0$ (d) $k_0 = 175 \cdot 2\pi/c_0$



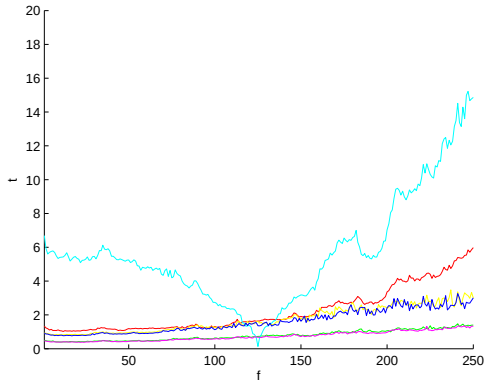
(e) $k_0 = 225 \cdot 2\pi/c_0$ (f) $k_0 = (25 + 50\lfloor f/50 \rfloor) \cdot 2\pi/c_0$

Figure B.5: Number of MATVECS per frequency for IDR(4), $h = L/100$.

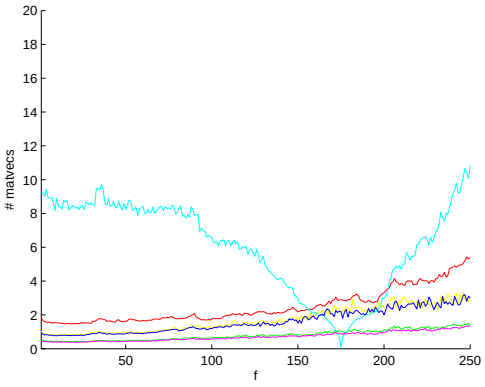


(a) $k_0 = 25 \cdot 2\pi/c_0$

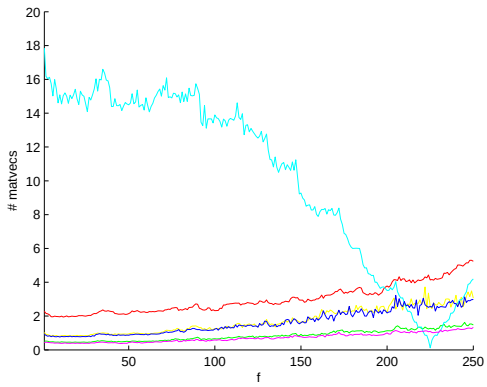
(b) $k_0 = 75 \cdot 2\pi/c_0$



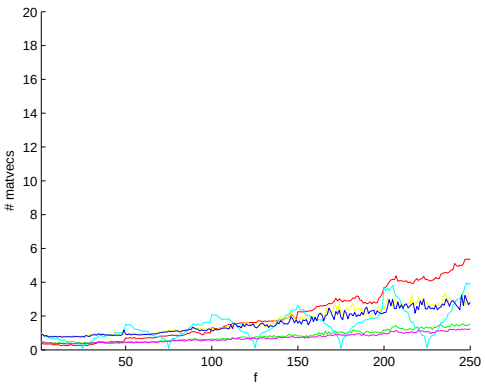
(c) $k_0 = 125 \cdot 2\pi/c_0$



(d) $k_0 = 175 \cdot 2\pi/c_0$



(e) $k_0 = 225 \cdot 2\pi/c_0$



(f) $k_0 = (25 + 50\lfloor f/50 \rfloor) \cdot 2\pi/c_0$

Figure B.6: Computation time (s) per frequency for IDR(4), $h = L/100$.

The efficiency of the preconditioners $\mathbf{P}^i(k_0)$ and $\mathbf{P}^r(k_0)$ combined with the incomplete LU factorisation seems to be independent of k_0. Both ILU(0) and ILUT(0.01) behave very much the same, no matter which preconditioner or what value of k_0 is chosen. So for this problem, the incomplete LU factorisations are not sensitive to the value of k_0.

Applying five preconditioners based on the frequency is useful for complete LU factorisation only and improves the results for $\mathbf{P}^r$ greatly. Still, the Laplace preconditioner $\mathbf{P}^i$ performs better in this case. Since we have to compute the LU factorisation five times, which is very time-consuming especially for large matrices, the shifted Laplace preconditioner $\mathbf{P}^i$ with a fixed value of k_0 seems to be a better choice for very large problems.

Bibliography

[1] W.E. Arnoldi. The principle of minimised iteration in the solution of the matrix eigenvalue problem. *Quart. Appl. Math.* 9:17-19, 1951.

[2] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, editors. *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*. SIAM, Philadelphia, 2000.

[3] R. Citarella, L. Federico, A. Cicatiello. Modal acoustic transfer vector approach in a FEM-BEM vibro-acoustic analyses. *Eng. Analyses with B.E.* 31:248-258, 2007.

[4] Y.A. Erlangga, C. W. Oosterlee, C. Vuik. *A Novel Multigrid Based Preconditioner For Heterogeneous Helmholtz Problems*. Department of Applied Mathematical Analysis, Delft, 2004.

[5] R. Fletcher. Conjugate gradient methods for indefinite systems. *Lecture Notes in Math.* 506:73-89 1976.

[6] R.W. Freund, N.M. Nachtingal. QMR: A quasi-minimal residual method for non-Hermitian linear systems. *Numer. Math.* 60:315-339, 1991.

[7] M.B. van Gijzen, Y.A. Erlangga, C. Vuik. Spectral analyses of the discrete Helmholtz operator preconditioned with a shifted Laplacian. *SIAM J. Sci. Comput.* 5:1942-1958, 2007.

[8] M.B. van Gijzen, G.L.G. Sleijpen, J.P.M. Zemke. *Flexible and multi-shift induced dimension reduction algorithms for solving large sparse linear systems*. Department of Applied Mathematical Analysis, Delft, 2011.

[9] M.B. van Gijzen, P. Sonneveld. An elegant IDR(s) variant that efficiently exploits bi-orthogonality properties. *TOMS* 1:5, 2011.

[10] M.H. Gutknecht, J.M. Zemke. Eigenvalue computations based on IDR. Res.Rep.SAM, Bericht TUHH 2010.

[11] N.J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, Philadelphia, 1996.

[12] B. Jegerlehner. *Krylov space solvers for shifted linear systems*. Department of Physics, Indiana University, 1996.

[13] J.J.I.M. van Kan, A. Segal, F.J. Vermolen. *Numerical methods in scientific computing*. VSSD, Delft, 2005.

[14] V. Mehrmann, C. Schröder. Nonlinear eigenvalue and frequency response problems in industrial practice. *J. Math. Ind.* 1:7, 2011.

[15] J.A. Meijerink, H.A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Math. Comp.* 31:148-163 1977.

[16] C.W. Oosterlee, C. Vuik. *Scientific computing*. Delft Institute of Applied Mathematics, Delft, 2009.

[17] A.D. Pierce. *Acoustics, An introduction to its physical principles and applications*. Acoustical Society of America, New York, 1989.

[18] S.W. Rienstra, A. Hirschberg. *An introduction to acoustics*. University of Technology, Eindhoven, 2004.

[19] Y. Saad, M.H. Schultz. GMRES: A generalised minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.* 7:856-869, 1986.

[20] Y. Saad. ILUT: A dual threshold incomplete ILU factorisation. *Numerical Linear Algebra Appl.* 1:387-402, 1994.

[21] Y. Saad. *Iterative methods for sparse linear systems*. Society for Industrial and Applied Mathematics, Philadelphia, 2003.

[22] V. Simoncini, D.B. Szyld. Interpreting IDR as a Petrov-Galerkin method. *SIAM J. Sci. Comput.* 32:1898-1912, 2010.

[23] G.L.G. Sleijpen, H.A. van der Vorst. Maintaining convergence properties of BiCGstab methods in finite precision arithmetic. *Numerical Algorithms* 10:203-223, 1995.

[24] V.I. Smirnov, N.A. Lebedev. *Functions of a complex variable: Constructive theory*. Translated from the Russian by Scripta Technica Ltd, The M.I.T. Press, Cambridge, Mass., 1968.

[25] P. Sonneveld, M.B. van Gijzen. IDR(s): A family of simple and fast algorithms for solving large nonsymmetric systems of linear equations. *SIAM J. Sci. Comput.* 31:1035-1062, 2008.

[26] P. Sonneveld. A fast Lanczos-type Solver for Nonsymmetric systems. *SIAM J. Sci. Comput.* 31(2):1025-1062 1989.

[27] P. Sonneveld. *On the convergence behaviour of IDR(s)*. Department of Applied Mathematical Analysis, Delft, 2010.

[28] M.J. Todd, E.A. Yildirim. On Khachiyans Algorithm for the Computation of Minimum Volume Enclosing Ellipsoids. *Discrete Appl. Math.* 155:1731-1744, 2007.

[29] R.S. Varga. *Gershgorin and His Circles*. Springer, Berlin, 2004.

[30] H.A. van der Vorst. Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.* 13:631-644 1992.

[31] P. Wesseling, P. Sonneveld. Numerical experiments with a multiple grid and a preconditioned Lanczos type method. *Approximation Methods for Navier-Stokes Problems, Lecture Notes in Math.* 543-562, 1980.