ANALYSIS OF CYCLIC REDUCTION FOR THE NUMERICAL SOLUTION OF THREE-DIMENSIONAL CONVECTION-DIFFUSION EQUATIONS

by

CHEN GREIF

B.Sc. (Applied Mathematics), Tel Aviv University, 1991
M.Sc. (Applied Mathematics), Tel Aviv University, 1994

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

THE FACULTY OF GRADUATE STUDIES

Department of Mathematics
Institute of Applied Mathematics

We accept this thesis as conforming to the required standard

........................................

........................................

........................................

........................................

........................................

THE UNIVERSITY OF BRITISH COLUMBIA

April 1998

© Chen Greif, 1998
In presenting this thesis in partial fulfillment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

(Signature) ____________________________

Department of Mathematics
The University of British Columbia
Vancouver, Canada

Date _________________________________
Abstract

This thesis deals with the numerical solution of convection-diffusion equations. In particular, the focus is on the analysis of applying one step of cyclic reduction to linear systems of equations which arise from finite difference discretization of steady-state three-dimensional convection-diffusion equations. The method is based on decoupling the unknowns and solving the resulting smaller linear systems using iterative methods. In three dimensions this procedure results in some loss of sparsity, compared to lower dimensions. Nevertheless, the resulting linear system has excellent numerical properties, is generally better conditioned than the original system, and gives rise to faster convergence of iterative solvers, and convergence in cases where solvers of the original system of equations fail to converge.

The thesis starts with an overview of the equations that are solved and general properties of the resulting linear systems. Then, the unsymmetric discrete operator is derived and the structure of the cyclically reduced linear system is described. Several important aspects are analyzed in detail. The issue of orderings is addressed and a highly effective ordering strategy is presented. The complicated sparsity pattern of the matrix requires careful analysis; comprehensive convergence analysis for block stationary methods is provided, and the bounds on convergence rates are shown to be very tight. The computational work required to perform cyclic reduction and compute the solution of the linear system is discussed at length. Preconditioning techniques and various iterative solvers are considered.
# Table of Contents

Abstract ii  
Table of Contents iii  
List of Tables v  
List of Figures vi  
Acknowledgements viii  

Chapter 1. Introduction 1  
1.1 Background ................................................................. 2  
1.1.1 The Convection-Diffusion Equation .................................. 2  
1.1.2 Finite Difference Methods ............................................. 4  
1.1.3 Solving the Linear System ............................................. 9  
1.2 One Step of Cyclic Reduction ........................................... 14  
1.2.1 The One-Dimensional Case ............................................ 14  
1.2.2 More Space Dimensions ............................................... 20  
1.3 Thesis Outline and Notation ............................................ 23  

Chapter 2. Cyclic Reduction 25  
2.1 Complete Cyclic Reduction .............................................. 26  
2.2 The Three-Dimensional Cyclically Reduced Operator .................... 29  
2.2.1 The Constant Coefficient Model Problem ................................. 31  
2.2.2 The Variable Coefficient Case ....................................... 36  
2.3 Properties of the Reduced Matrix ....................................... 39  

Chapter 3. Ordering Strategies 46  
3.1 Block Ordering Strategies for 3D Grids ................................ 48  
3.2 The Family of Two-Plane Orderings ................................... 50  
3.3 The Family of Two-Line Orderings .................................... 54  
3.4 Comparison Results ...................................................... 58  

Chapter 4. Convergence Analysis for Block Stationary Methods 64  
4.1 Symmetrization of the Reduced System .................................. 64  
4.1.1 The Constant Coefficient Case .......................................... 65  
4.1.2 The Variable Coefficient Case ......................................... 72  
4.2 Bounds on Convergence Rates ........................................... 75  
4.3 "Near-Property A" for 1D Partitioning of the Two-Plane Matrix ............. 87  
4.4 Computational Work ...................................................... 94  
4.5 Comparison with the Unreduced System .................................. 97  
4.6 Fourier Analysis .......................................................... 103  
4.7 Comparisons ............................................................... 107
# Table of Contents

## Chapter 5. Solvers, Preconditioners, Implementation and Performance 113

5.1 Krylov Subspace Solvers and Preconditioners - Overview .......................... 113  
5.2 Incomplete Factorizations for the Reduced System ................................. 120  
5.3 The Overall Cost of Construction of the Reduced System ......................... 126  
5.4 Numerical Experiments ........................................................................... 128  
5.5 Vector and Parallel Implementation ...................................................... 140

## Chapter 6. Summary, Conclusions and Future Research 153

6.1 Summary and Conclusions ........................................................................ 153  
6.2 Future Research ...................................................................................... 157

Bibliography 160
List of Tables

4.1 comparison between the computed spectral radius and the bound for the 1D splitting 86
4.2 comparison between the computed spectral radius and the bound for the 2D splitting 86
4.3 comparison of computed spectral radii of the 1D Jacobi iteration matrix with matrices which are consistently ordered ............................................................... 92
4.4 comparison between iteration counts for the reduced and unreduced systems ...... 108
4.5 iteration counts for different iterative schemes ........................................... 110
4.6 iteration counts for one nonzero convective term ....................................... 111
4.7 iteration counts for two nonzero convective terms ...................................... 111

5.1 number of nonzero elements in the ILU(1) factorization .......................... 123
5.2 computational work involved in the construction of the reduced matrix .......... 127
5.3 comparison between the computed spectral radii and the bounds .................... 129
5.4 comparison of solving work/time for various mesh sizes .............................. 130
5.5 comparison between estimated condition numbers ...................................... 132
5.6 performance of block stationary methods for Test Problem 1 ....................... 132
5.7 construction time/work of ILU preconditioners ........................................ 133
5.8 performance of QMR for Test Problem 1 .................................................. 134
5.9 performance of BiCG for Test Problem 1 .................................................. 134
5.10 performance of Bi-CGSTAB for Test Problem 1 ........................................ 135
5.11 performance of CGS for Test Problem 1 .................................................. 135
5.12 overall flop counts and average computed constants for Test Problem 2 ........ 138
5.13 performance of various incomplete factorizations for Test Problem 2 ............. 139
5.14 norm of the error for Test Problem 3 ...................................................... 140
5.15 comparison of performance for Test Problem 3 ......................................... 141
5.16 comparison of iteration counts of natural and multicolor orderings .............. 152
List of Figures

1.1 computational molecules ......................................................... 8
1.2 numerical solution of Eq. (1.24) ................................................... 14
1.3 red/black ordering of the one-dimensional grid ................................. 15
1.4 the matrix associated with the red/black ordering for the one-dimensional case. 16
1.5 eigenvalues of the reduced matrix (solid line) and the unreduced matrix (broken line) for the one-dimensional model problem with n = 65 and β = 0.3. ....................... 19
1.6 natural lexicographic ordering of the tensor-product grids. .................. 21
1.7 sparsity patterns of the matrices. ............................................... 22
1.8 red/black ordering in the two-dimensional case. ................................ 22
2.1 a three-dimensional checkerboard. ............................................... 30
2.2 red/black ordering of the 3D grid ............................................... 31
2.3 points that are affected by the block Gaussian elimination .................... 32
2.4 structure of the computational molecule associated with the reduced operator ...... 33
2.5 sparsity pattern of the lexicographically ordered reduced matrix .............. 35
2.6 eigenvalues of both systems for Poisson’s equation ................................ 43
2.7 singular values of both matrices for Eq. (2.24) ................................ 44
3.1 orderings of the 2D block grid .................................................. 49
3.2 three members in the family of natural two-plane orderings ................. 50
3.3 red/black and toroidal two-plane ordering corresponding to x-y oriented 2D blocks . 51
3.4 sparsity patterns of two members of the two-plane family of orderings ...... 53
3.5 block computational molecule corresponding to the family of orderings 2PN ........ 53
3.6 four-color 1D block ordering .................................................. 54
3.7 ordering and sparsity pattern of the matrix associated with 2LNxy ordering .... 55
3.8 block computational molecule corresponding to the ordering strategy 2LNxy ........ 58
3.9 possible block partitions of the two-plane matrix. ................................ 59
3.10 a zoom on 2D blocks ................................................................... 60
3.11 comparison of the spectral radii of block Jacobi iteration matrices for cross-sections of the mesh Reynolds numbers ............................................. 61
3.12 spectral radii of iteration matrices vs. mesh Reynolds numbers .......... 62
3.13 symmetric reverse Cuthill-McKee ordering of the reduced matrix .......... 63
4.1 sparsity patterns of the matrices involved in the proof of Lemma 4.8 ........... 84
4.2 “Near Property A” for the 1D splitting ......................................... 89
4.3 sparsity pattern of the matrix C_{1}^{(1)} ......................................... 90
4.4 the function h_{B}(α) .................................................................. 93
4.5 spectral radius of the SOR iteration matrix vs. the relaxation parameter .... 95
4.6 comparison of the spectral radii of the Gauss-Seidel iteration matrices ....... 103
5.1 a two-dimensional slice of the stencil associated with U ....................... 121
5.2 fill-in in the construction of ILU(1) in the plane containing the gridpoint for which the discretization was done ........................................... 121
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3</td>
<td>fill-in in the construction of ILU(1) in the plane adjacent to the gridpoint for which the discretization was done</td>
<td>122</td>
</tr>
<tr>
<td>5.4</td>
<td>sparsity patterns of the factors of ILU(1) and ILU(2)</td>
<td>123</td>
</tr>
<tr>
<td>5.5</td>
<td>sparsity patterns of factors for ILU with drop tolerance $10^{-2}$</td>
<td>125</td>
</tr>
<tr>
<td>5.6</td>
<td>$l_2$-norm of relative residual for preconditioned Bi-CGSTAB</td>
<td>131</td>
</tr>
<tr>
<td>5.7</td>
<td>a 2D slice of the numerical solution of Test Problem 3</td>
<td>140</td>
</tr>
<tr>
<td>5.8</td>
<td>2D mesh architecture $(3 \times 3)$</td>
<td>143</td>
</tr>
<tr>
<td>5.9</td>
<td>a 2D slice of gridpoints associated with one processor</td>
<td>145</td>
</tr>
<tr>
<td>5.10</td>
<td>the part of the reduced matrix that contains the gridpoints associated with a certain subcube, and the rectangular matrix before local ordering</td>
<td>149</td>
</tr>
<tr>
<td>5.11</td>
<td>the “local part” of the rectangular matrix after re-ordering</td>
<td>150</td>
</tr>
<tr>
<td>5.12</td>
<td>a 2D slice of processors that hold entries of 1D or 2D sets of unknowns</td>
<td>151</td>
</tr>
</tbody>
</table>
Acknowledgements

I would like to thank my supervisor, Jim Varah, for his devoted guidance. Jim has been of
great inspiration to me, always supportive, enthusiastic, insightful and helpful. I learned a lot
from Jim about matrix analysis and linear algebra, and am deeply grateful to him for the many
hours he devoted to me, and for making these years of work on my thesis a most pleasant,
enjoyable and satisfying experience.

I am indebted to Gene Golub. This work started after I had read Gene’s joint papers with
Howard Elman, and was fascinated by the results and the analysis. I am grateful to Gene for
his interest in my work, for pointing out useful references and important aspects of the problem,
and for his kind hospitality during my visits at Stanford University. I would also like to thank
Howard Elman for being the external examiner of this thesis, and in particular, for his valuable
comments and suggestions concerning Section 4.3, which substantially improved the exposition.

Many thanks to Uri Ascher. Uri has been very helpful, and I am grateful to him for
numerous pleasant and fruitful discussions and for teaching me many interesting and important
aspects of numerical computation.

Thanks to Michael Ward and Anthony Peirce for their interest and input in our committee
meetings; thanks to Michael for providing interesting problems and references. Thanks also
to Eldad Haber, for many stimulating discussions on everything from global politics to cyclic
reduction. I have benefited from several discussions with Xiaodi Sun and Xiao-Wen Chang and
thank them for their helpful suggestions.

Finally, very special thanks to my wife Roni and my daughter Shakaed, who have been a
wonderful source of motivation and support, and have given me all the love and encouragement
I could hope for.
Chapter 1
Introduction

Mathematical models for fluid mechanics are of major interest in science and engineering. The important task of modelling physical phenomena (such as the motion of particles in a fluid or the relative motion between an object and a fluid) is complicated and challenging, not only from the point of view of the scientists who design the model for a given problem, but also from the point of view of the numerical analysts who are interested in finding accurate numerical solutions for the model.

Concentration of a pollutant (for example radioactive waste or atmospheric pollution), flow of fluid past an obstacle, the motion of airplanes and submarines, the wind that blows past a bridge, semi-conductor modelling, and even some financial models of share options - all these important problems have in common the type of equations that are used to describe them. These are partial differential equations whose associated unknown quantities are typically functions of space and time; the equations are typically second order in space and first order in time; they could be nonlinear, and an analytical solution is usually not available, due to their complicated nature. One such family of equations, which we will focus on throughout this thesis, is known by the name convection-diffusion equations. A great deal of investigation and analysis has been performed on these equations in the last few decades. They encapsulate many mathematical difficulties, and are of major importance in several areas of application. In the preface to Morton’s book [81] on numerical solution of this type of equations, the first sentence reads: “Accurate modelling of the interaction between convective and diffusive processes is the most ubiquitous and challenging task in the numerical approximation of partial differential
Chapter 1. Introduction

In this thesis the focus is on the particular problem of how to solve linear systems that arise from finite difference discretization of three-dimensional steady-state convection-diffusion equations. In particular, we analyze the technique of one step of cyclic reduction, which proves to be numerically stable and effective. The description of the method and its analysis are to come later. We first give a general description of the equations we deal with, and the linear systems that arise when the equations are discretized.

1.1 Background

1.1.1 The Convection-Diffusion Equation

We consider steady-state convection-diffusion equations of the following type:

\begin{align}
-\nabla \cdot (D \nabla u) + \nabla \cdot (E u) &= F \quad \text{on } \Omega ; \\
\nu u + \varphi \frac{\partial u}{\partial n} &= G \quad \text{on } \partial \Omega .
\end{align}

Here \( u \) is an unknown quantity. We focus on the three-dimensional problem, in which case \( u, D, E, F \) and \( G \) are trivariate functions, and \( \Omega \in \mathbb{R}^3 \).

Consider for example the equations associated with fluid motion. Newton’s second law of motion for fluids, according to which the rate of change of momentum of a fluid particle is equal to the net force acting on it, is represented by the \textit{Navier-Stokes} equations (see, for example, [46]). Considering the particular class of equations for \textit{viscous, incompressible} fluids, the equations for this type are:

\begin{align}
\rho (u_t + u \cdot \nabla u) &= -\nabla p + \mu \Delta u + f ; \\
\nabla \cdot u &= 0 ,
\end{align}

2
where:

- $\rho$ is the density of the fluid.

- The unknown quantity $u$ is the velocity field: a vector function, each of whose elements depends on the spatial variables $x$, $y$, and $z$, and on the time variable, $t$.

- $p$ is the pressure: a scalar quantity considered to be unknown, which depends on the spatial variables and the time.

- $\mu$ is the coefficient of viscosity, which is a physical property of the material.

- $f$ denotes the external force acting on the fluid, which is assumed to be known. Such a force could be, for example, gravity.

Eq. (1.2a) is called the momentum equation. Eq. (1.2b) is the continuity equation, and illustrates the incompressibility of the flow, which means that pressure variations do not produce any significant density variations. The most obvious difficulty associated with the Navier-Stokes equations is the nonlinearity. One can deal with it in several ways. For example, using Picard-like iteration [73] for the momentum equation leads to the linear Oseen equation [44], which for steady-state is a linear equation of the form (1.1). Other common ways to handle the nonlinearity are Newton’s method or quasi-Newton methods [86]. Typically, in a nonlinear iterative solver, at each iterate a linear system of equations has to be solved. It is therefore of much importance to derive efficient methods for numerically solving these linear systems.

Equations of the type (1.1) arise when considering, for example, temperature variation, or varying concentration of some substance (e.g., chemical or radioactive pollutant) mixed with the fluid. Related examples are thermal processes in meteorology, salt concentration in oceans, and air and water pollution [109]. An important term here is “convection”, which describes heat transfer in case of temperature variation, or mass transfer when concentration variation is considered. The distribution of the concentration $c$ of a fluid is determined by its advection by moving particles and by its diffusion between fluid particles. The equation is

$$c_t + u \cdot \nabla c = \nabla \cdot (D \nabla c) + f,$$  

(1.3)
Chapter 1. Introduction

Here \( u \) is the velocity and \( D \) is a coefficient of diffusivity. If these quantities do not depend on \( c \), Eq. (1.3) is linear. This is a typical convection-diffusion model problem.

In order to “quantify” difficulties that arise in physical problems such as the ones that have been described above, a dimensionless quantity that is typical of the problem is defined. In the problem describing concentration of a pollutant, it is the Péclet number, which is the velocity times the length scale of the domain, divided by the diffusivity coefficient. In the Navier-Stokes equations, the dimensionless Reynolds number is the product of velocity, density and length, divided by the viscosity of the fluid. Reynolds number is denoted by \( Re \); Péclet number is denoted by \( Pe \). These numbers significantly vary from one problem to another. As it turns out, as long as these numbers are small, the problem is relatively simple to solve and theory can provide an accurate description of the physical phenomenon. For example, according to [109], the flow past a cylinder can be quite accurately modelled for Reynolds numbers up to about 40; the flow is called laminar in this case. As the Reynolds numbers pass this limit, there is separation to what is called turbulent flow, instability develops, and it is much more difficult to predict or model the behavior of the flow.

The physical problems scientists and engineers are interested in are obviously not limited to low Reynolds or Péclet numbers. When these numbers are high, the difficulties that arise are not only limited to the physical sense; there are also difficulties in the numerical solution of the equations.

1.1.2 Finite Difference Methods

There are various approaches for numerical solution of convection-diffusion equations. Among them we mention finite differences[80],[91],[101], finite elements[17],[66],[72], and finite volumes [70]. The approach used in this thesis is finite differences. Here the idea is to approximate the differential operator by a difference quotient which can be thought of as a discrete difference operator. For example, consider the ordinary differential equation

\[
\mathcal{L}u = f
\]  
(1.4)
Chapter 1. Introduction

on the interval \((0, 1)\), where \(\mathcal{L}\) is a differential operator of the type \(\mathcal{L} = a(x) \frac{d^2}{dx^2} + b(x) \frac{d}{dx}\), and either \(u\) or \(u'\) are given on \(x = 0\) and \(x = 1\). First, the problem is discretized as follows: a grid consisting of \(n\) gridpoints is defined, each of which is a point where an approximate solution of \(u\) is to be computed. Suppose the grid is uniform, denote its size by \(h = \frac{1}{n+1}\), and let \(u_i\) denote the numerical solution at \(x_i = ih, \ i = 1, \ldots, n\). If \(h\) is small enough and \(u\) is sufficiently smooth [101], the first few terms of the Taylor expansion can be used to derive an effective approximation; given \(i, u(x_{i+1})\) can be expressed as:

\[
u(x_{i+1}) = u(x_i) + h u'(x_i) + \frac{h^2}{2} u''(x_i) + \frac{h^3}{3!} u'''(x_i) + O(h^4) . \tag{1.5}\]

Similarly, for \(u(x_{i-1})\):

\[
u(x_{i-1}) = u(x_i) - h u'(x_i) + \frac{h^2}{2} u''(x_i) - \frac{h^3}{3!} u'''(x_i) + O(h^4) . \tag{1.6}\]

Now, simple combinations of Eq. (1.5) and (1.6) provide a few possible approximation formulas for \(u'(x_i)\) and \(u''(x_i)\), expressed in terms of \(u(x_{i-1}), u(x_i)\) and \(u(x_{i+1})\). For the first derivative it is straightforward to verify that:

- \(u'(x_i) = \frac{u(x_{i+1}) - u(x_{i-1})}{h} + O(h) .\)
- \(u'(x_i) = \frac{u(x_{i+1}) - u(x_i)}{h} + O(h) .\)
- \(u'(x_i) = \frac{u(x_{i+1}) - u(x_{i-1})}{2h} + O(h^2) .\)

The following approximation schemes can therefore be defined:

1. First order accurate “forward scheme”:

\[
(u')_i = \frac{u_{i+1} - u_i}{h} .\tag{1.7}
\]

2. First order accurate “backward scheme”:

\[
(u')_i = \frac{u_i - u_{i-1}}{h} .\tag{1.8}
\]

3. Second order accurate “centered scheme”:

\[
(u')_i = \frac{u_{i+1} - u_{i-1}}{2h} .\tag{1.9}
\]
Chapter 1. Introduction

For the second derivative, one usually uses a second order accurate centered scheme, which can be obtained by summing Eq. (1.5) and Eq. (1.6):

\[
(u''_i) = \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}.
\] (1.10)

What all the approximation schemes that have been presented have in common, is that the derivatives of \( u \) at the point \( x_i \) are approximated by three values only, namely \( u_{i-1}, u_i \) and \( u_{i+1} \). The difference operator is thus a three-point operator.

The generalization to more space dimensions is straightforward. In the two-dimensional case, if \( n \) is the number of unknowns in each horizontal or vertical line of the mesh and \( h = \frac{1}{n+1} \) is the grid size, we denote by \( \{u_{i,j}\} \) the numerical approximations to \( u(ih, jh), \ 1 \leq i, j \leq n \), and discretize the differential equation as follows [101]:

- The first order derivatives are approximated by either a second order accurate centered difference scheme, or by a first order accurate one sided scheme. For example, for the derivative in \( x \)-direction, in case of centered differences we have

\[
\frac{\partial}{\partial x} u(ih, jh) \approx \frac{u_{i+1,j} - u_{i-1,j}}{2h};
\] (1.11)

a backward scheme would be

\[
\frac{\partial}{\partial x} u(ih, jh) \approx \frac{u_{i,j} - u_{i-1,j}}{h};
\] (1.12)

and a forward scheme is

\[
\frac{\partial}{\partial x} u(ih, jh) \approx \frac{u_{i+1,j} - u_{i,j}}{h}.
\] (1.13)

- The second derivatives are approximated using a second order accurate centered difference scheme. For example:

\[
\frac{\partial^2}{\partial x^2} u(ih, jh) \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2},
\] (1.14)

and analogously for the \( y \)-derivative.
Chapter 1. Introduction

At this point the pattern (for any number of space dimensions) is clear: each gridpoint is associated with its two neighbors in each direction, and thus: in the two-dimensional case five points are involved in the difference equation, and in the three-dimensional case seven points are involved. Indeed, for the 2D and 3D case, the operators are termed the five-point operator and the seven-point operator respectively [67]. The associated computational molecule is defined as the set of values that are associated with the discretization of a certain gridpoint. In the (general) variable coefficient case the values of the computational molecule are gridpoint-dependent. On the other hand, in the constant coefficient case the computational molecule has the same form for all gridpoints that are not next to the boundary.

Throughout the thesis, we shall use the term “the model problem” in reference to the following constant coefficient equation:

\[-\Delta u + V^T \nabla u = w,\]  

(1.15)
on \Omega = (0, 1)^s, subject to Dirichlet boundary conditions on \(\partial \Omega\). Here \(s\) is the number of space dimensions, and \(V^T\) is an \(s\)-dimensional vector of constants: \(V = (\sigma)\) if \(s = 1\), \(V = (\sigma, \tau)\) if \(s = 2\), and \(V = (\sigma, \tau, \mu)\) if \(s = 3\).

Let

\(\beta = \frac{\sigma h}{2}, \quad \gamma = \frac{\tau h}{2}, \quad \delta = \frac{\mu h}{2},\)  

(1.16)
Clearly, in two dimensions only \(\beta\) and \(\gamma\) are defined, and in one dimension only \(\beta\) is defined. \(\beta, \gamma\) and \(\delta\) are the mesh Reynolds numbers.\(^1\) In order to examine how effective a numerical solution would be, both the magnitude of the PDE coefficients and the size of the grid are considered, and a mesh Reynolds number encapsulates both of them in one mathematical quantity.

The discrete operators can be expressed in terms of the mesh Reynolds numbers. For the model problem, the computational molecules for interior gridpoints in one, two and three dimensions are graphically illustrated in Fig. 1.1. If centered differences are used for approximating

\(^{1}\text{We note that we are following the definition used by Elman & Golub in [40], [41], [42]. In some texts, e.g. [46], [81], there is no division by 2 in (1.16). The words “cell” instead of “mesh”, and “Péclet” rather than “Reynolds”, are used interchangeably in the literature and are equivalent.}\)
the first order derivatives, the values of the computational molecules are as follows:

\[
\begin{align*}
a &= 6; 
b &= -1 - \gamma; 
c &= -1 - \beta; 
d &= -1 + \beta; 
e &= -1 + \gamma; 
f &= -1 - \delta; 
g &= -1 + \delta.
\end{align*}
\]  
(1.17)

For backward approximations we have:

\[
\begin{align*}
a &= 6 + 2 \cdot (\beta + \gamma + \delta); 
b &= -1 - 2\gamma; 
c &= -1 - 2\beta; 
d &= -1; 
e &= -1; 
f &= -1 - 2\delta; 
g &= -1.
\end{align*}
\]  
(1.18)

For forward approximations we have:

\[
\begin{align*}
a &= 6 - 2 \cdot (\beta + \gamma + \delta); 
b &= -1; 
c &= -1; 
d &= -1 + 2\beta; 
e &= -1 + 2\gamma; 
f &= -1; 
g &= -1 + 2\delta.
\end{align*}
\]  
(1.19)

![Figure 1.1: computational molecules associated with the three-point, five-point and seven-point operators for the constant coefficient problem.](image)

The difference operators for the 1D, 2D and 3D problems, after scaling by $h^2$, are given respectively by

\[
\begin{align*}
F_1 u_{i,j,k} &= a u_{i,j,k} + c u_{i-1,j,k} + d u_{i+1,j,k}; \\
F_2 u_{i,j,k} &= a u_{i,j,k} + b u_{i,j-1,k} + c u_{i-1,j,k} + d u_{i+1,j,k} + e u_{i,j+1,k}; \\
F_3 u_{i,j,k} &= a u_{i,j,k} + b u_{i,j-1,k} + c u_{i-1,j,k} + d u_{i+1,j,k} + e u_{i,j+1,k} + f u_{i,j,k-1} + g u_{i,j,k+1}.
\end{align*}
\]  
(1.20a, 1.20b, 1.20c)
Chapter 1. Introduction

1.1.3 Solving the Linear System

Once the finite difference scheme has been determined, we have a large sparse linear system of equations for the unknowns. The sparsity pattern of the associated matrix depends on the ordering of the unknowns. For many commonly used orderings the matrix is narrow-banded. In more than one dimension the band is necessarily sparse, regardless of the ordering used.

Suppose the matrix is of size $N \times N$, and the linear system is given by

$$Ax = b.$$  \hspace{1cm} (1.21)

The direct solving approach of Gaussian elimination [54] is equivalent to transforming Eq. (1.21) into a system $Ux = y$ where $U$ is upper triangular, by solving a system $Ly = b$ where $L$ is a unit lower triangular matrix. $A = LU$ is known as the LU decomposition of $A$, and storing $L$ and $U$ is advantageous when one needs to solve a system with the same matrix but with multiple right-hand-side vectors. Unfortunately, for sparse matrices this approach suffers the major drawback of fill-in in the factors $L$ and $U$, which occurs during the factorization: the bandwidth of the matrix is preserved, but the sparsity inside the band is destroyed. The issue here is not only waste of storage; fill-in causes an unacceptable increase in computational work, and in fact, the amount of work increases with the number of space dimensions when finite difference discretization of partial differential equations is concerned. It is thus advisable to use iterative methods, which exploit the sparsity pattern of the matrix and typically use only a very small number of extra column vectors in addition to the storage required for the linear system. These are methods which generate a sequence of vectors $\{x^{(k)}\}$, which converge to the solution $x$. The amount of computational work involved in solving the system is modest when efficient iterative techniques are used. The multigrid method [16],[115], for example, can solve the system with only $O(N)$ floating point operations. A typical iterative method involves essentially only matrix-vector products (as opposed to direct methods). Throughout the thesis we shall consider stationary methods and Krylov subspace methods. Derivation and analysis of these methods can be found in many books. In particular, we mention the books of Varga [113] and Young [117] on stationary methods, the book of Greenbaum [61] on Krylov subspace
solvers and preconditioners, and the books of Golub & Van Loan [55] and Saad [95], which present a variety of topics, including stationary and Krylov subspace methods. In Chap. 5 a short overview on Krylov subspace solvers is to be given. Below we briefly describe stationary methods. This is a family of fixed point schemes, which have been known and used for a long time. Consider the system $Ax = b$ and denote by $x^{(k)}$ the approximation to the solution $x$ at the $k$th iterate. The idea is: given an initial guess $x^{(0)}$, iterate as follows:

$$Mx^{(k+1)} = (M - A)x^{(k)} + b, \quad k = 0, 1, \ldots$$

(1.22)

where $M$ is a matrix associated in some way with the matrix $A$. Ideally, $M$ should approximate $A$ (or rather, $M^{-1}$ should be an effective approximation to $A^{-1}$), and at the same time solving a system involving $M$ should be inexpensive - much cheaper than solving a system involving $A$. Clearly, these two requirements are somewhat contradictory, and here the main difficulty in picking an efficient scheme lies. Standard schemes are based on using a splitting [113] of the matrix $A$, which is the operation of writing it as

$$A = M - N.$$  

(1.23)

The matrix $M^{-1}N$ is the iteration matrix. A necessary condition for convergence for any initial guess is $\rho(M^{-1}N) < 1$, where $\rho$ denotes spectral radius (the maximal absolute value among the eigenvalues).

Denoting the diagonal part of $A$ by $D$ and its strict lower and upper triangular parts by $E$ and $F$, respectively, classical schemes are point Jacobi, which corresponds to picking $M = D$, and point Gauss-Seidel, which corresponds to picking $M = D + E$. The point Successive-Over-Relaxation scheme (SOR) is obtained by picking $M = \frac{\omega}{\omega_d}[D + \omega E]$ where $\omega$ is a scalar between 0 and 2, and can be considered as a technique for accelerating the convergence of the Gauss-Seidel scheme. When $\omega$ is close to the optimal value (that is, the value that would give the smallest possible spectral radius of $M^{-1}N$), the convergence of the SOR scheme is substantially faster than the convergence of Jacobi and Gauss-Seidel. A great deal of convergence analysis for the SOR method (mainly for symmetric positive definite matrices) has been done by Young in the 1950s and 1960s. In particular, Young defined a class of “consistently ordered” matrices, for
which strong connections between the eigenvalues of the Jacobi, Gauss-Seidel and SOR iteration matrices exist. Some of these results will be discussed in Chapter 4. Another technique for accelerating the convergence of iterative schemes is the Chebyshev method (see Golub & Varga [59]).

Other splittings which are more relevant to this work, are block splittings. The idea is to work with square submatrices of the system’s matrix rather than single matrix elements, as the basic building blocks for the splitting. For example, the block Jacobi scheme corresponds to taking $M = D$, where $D$ now is no longer the diagonal part of $A$; rather, it is the block-diagonal part of $A$. General convergence results for stationary methods are given in Varga’s book [113]. Varga presents many interesting and powerful results, and below we mention the ones which will be useful for us later. See [113] for proofs.

**Definition 1.1.** A matrix $B = (b_{i,j})$ is reducible if there exists a permutation matrix $P$ such that $P^T BP$ is a $2 \times 2$ block upper triangular matrix.

If a matrix does not satisfy the conditions of Defn. 1.1, it is termed irreducible.

**Definition 1.2.** A matrix $B$ is called positive (nonnegative) if all its elements are positive (nonnegative), and is denoted by $B > 0$ ($B \geq 0$).

An important class of matrices is the following:

**Definition 1.3.** A matrix $B = (b_{i,j})$ is an M-matrix if it is nonsingular, $b_{i,j} \leq 0$ for all $i \neq j$, and $B^{-1} \geq 0$.

Obviously, one does not want to compute the inverse of a matrix in order to determine whether or not it is an M-matrix. A useful way of determining whether a matrix is an M-matrix is:

**Theorem 1.1.** [113, Cor. 1, p. 85] If a matrix $B = (b_{i,j})$ is a real, irreducibly diagonally dominant $n \times n$ matrix with $b_{i,j} \leq 0$ for all $i \neq j$ and $b_{i,i} > 0$ for all $1 \leq i \leq n$, then $B^{-1} > 0$. 

11
Chapter 1. Introduction

As it turns out, the above-defined properties could in some circumstances guarantee convergence of stationary schemes:

**Definition 1.4.** $A = M - N$ is a regular splitting of $A$ if $M$ is nonsingular with $M^{-1} \geq 0$ and $N \geq 0$.

**Theorem 1.2.** [113, Thm. 3.13, p. 89] If $A = M - N$ is a regular splitting of $A$ and $A^{-1} \geq 0$, then the iterative scheme associated with this splitting converges for any initial guess.

When attempting to solve the linear systems that arise from discretizing PDEs, potential difficulties that might arise are:

1. If any of the mesh Reynolds numbers is nonzero, the matrix is nonsymmetric. The nonsymmetry complicates the analysis significantly. The numerical properties are difficult to assess, compared to the symmetric case for which a great deal of analysis is available. The matrix is not necessarily diagonally dominant.

2. If the Reynolds numbers are large the matrix can be ill-conditioned, leading to reduced accuracy in the numerical solution and possible divergence of iterative methods.

A good example of these numerical difficulties is when the original (continuous) problem has ill-conditioning which the discretized linear system inherits. This occurs, for example, when considering the singularly perturbed exit problem [77] of Brownian motion of particles confined by a finite potential well. In this case the continuous eigenvalue problem has an eigenvalue which is exponentially small, and a standard finite difference method would result in a matrix which is close to singular. For discussion and examples, see Sun [103] and references therein.

It is important to distinguish between numerical difficulties that arise as a result of the discretization scheme itself, and ones that arise when coming to solve the underlying linear system. As an illustration of this, consider the following classical problem (see, for example, [46], [81], [97]):

$$-\varepsilon u''(x) + \sigma u' = 0 \quad , \quad u(0) = 0, \; u(1) = 1 \quad .$$

(1.24)
Chapter 1. Introduction

This equation has the exact solution

$$u(x) = \frac{e^{x/x} - 1}{e^{x/x} - 1}.$$  \hspace{1cm} (1.25)

Here the mesh Reynolds number is $\beta = \frac{\sigma h}{2\varepsilon}$. Using centered difference schemes for discretizing $u''$ and $u'$, the corresponding difference equation is given by

$$(-1 - \beta)u_{j-1} + 2u_j + (-1 + \beta)u_{j+1} = 0.$$  \hspace{1cm} (1.26)

In this simple case, the difference equation can be solved directly by seeking a solution of the form $u_j = \varphi^j$, and we get a quadratic equation whose solution for $\varphi$ is [46]:

$$\varphi_+ = \frac{1 + \beta}{1 - \beta} \quad ; \quad \varphi_- = 1.$$  \hspace{1cm} (1.27)

After incorporating the boundary conditions, we obtain:

$$u_j = \frac{\varphi_j^1 - 1}{\varphi_+^{n+1} - 1}.$$  \hspace{1cm} (1.28)

When $\beta > 1$ the numerical solution is oscillatory ($u_j$ changes sign in dependence on the parity of $j$), whereas the analytical solution of the differential equation is smoothly monotonically increasing. The larger $\beta$ is, the more oscillatory and less accurate (as an approximation to the analytical solution) the numerical solution is. In Fig. 1.2(a) the oscillations are illustrated. In this graph, $\sigma$ was chosen to be equal to 1, $\varepsilon = 0.01$, and a uniform 16-point grid was used. The solid line represents the analytical solution, and the broken line represents the numerical solution.

Here the oscillations are not a result of numerical instability in the solution process; rather, they are a result of the finite difference scheme that is used. The natural remedy in this case would be to use a different numerical scheme. For example, if upwind differences are used, then the numerical solution is smooth - see Fig. 1.2(b). However, in this case the scheme is only first order accurate, and in particular, the numerical solution is not accurate at the vicinity of the boundary layer.
1.2 One Step of Cyclic Reduction

At this point we present the technique that this thesis is concerned with: (one step of) cyclic reduction. The idea of the method is to take the linear system arising from a discretization scheme, decouple the unknowns, and then compute the solution by solving smaller systems. The smaller systems generally have better numerical properties compared to the original system (including a smaller condition number), and thus by performing this procedure the difficulties described in the previous section become easier to handle.

1.2.1 The One-Dimensional Case

In order to illustrate cyclic reduction and its advantages, we first consider a simple one-dimensional model problem:

\[-u''(x) + \sigma u' = 0\ , \tag{1.29}\]

subject to Dirichlet boundary conditions. The most common ordering of the unknowns is the natural lexicographic ordering. In one dimension this ordering simply means that for each \(i, i = 1, \ldots, n\), the \(i\)th unknown corresponds to the numerical solution approximating \(u(ih)\).

Let \(\beta = \frac{\sigma h}{2}\) be the mesh Reynolds number. The matrix associated with the linear system,
using centered difference discretization, is \([40]\):

\[
A = \begin{pmatrix}
2 & -1 + \beta \\
-1 - \beta & 2 & -1 + \beta \\
0 & -1 - \beta & 2 & -1 + \beta \\
& \ddots & \ddots & \ddots \\
& & \ddots & \ddots & \ddots \\
& & & -1 - \beta & 2 & -1 + \beta \\
& & & 0 & -1 - \beta & 2 \\
\end{pmatrix}
\] \quad (1.30)

Lexicographic ordering is only one alternative. Another common ordering strategy is the red/black ordering: referring to the grid as a checkerboard, each gridpoint is assigned a color, either red or black, and points of one of the colors, say red, are numbered first. In the one-dimensional case red/black ordering means that we take the indices used in the lexicographic ordering and re-order them so that the odd-numbered indices are ordered first. This is illustrated in Fig. 1.3.

Figure 1.3: red/black ordering of the one-dimensional grid

The sparsity pattern of the matrix associated with the red/black ordering is depicted in Fig. 1.4. It is evident by looking at the matrix in this figure that the linear system has the form:

\[
\begin{pmatrix}
B & C \\
D & E
\end{pmatrix}
\begin{pmatrix}
q^{(r)} \\
q^{(b)}
\end{pmatrix}
= 
\begin{pmatrix}
w^{(r)} \\
w^{(b)}
\end{pmatrix}
\] \quad (1.31)

where both \(B\) and \(E\) are diagonal. In Eq. (1.31) the superscripts \((r)\) and \((b)\) have been introduced to illustrate the dependence on the color of the gridpoint. Now, referring to the
Figure 1.4: the matrix associated with the red/black ordering for the one-dimensional case.

matrix as a $2 \times 2$ block matrix, a simple process of block Gaussian elimination in Eq. (1.31) leads to a system whose second block-row is a (smaller) system for the black points, which is called the reduced system [40]:

$$\begin{bmatrix} E & DB \\ \end{bmatrix} u^{(i)} = w^{(i)} - DB^{-1} w^{(r)}. \tag{1.32}$$

Since $B^{-1}$ is diagonal and has entries whose typical values are not zero or close to zero (when an appropriate discretization scheme is used), the inversion is numerically stable. The new system in the 1D case is tridiagonal, just like the original system, but is half the size. Once the solution for the black points is computed, the solution for the red points corresponds to solving a diagonal system. The procedure of moving from system (1.31) to system (1.32) amounts to performing one step of cyclic reduction.

In order to illustrate the advantages of one step of cyclic reduction in the nonsymmetric case, let us refer back to Eq. (1.30). The region of numerical stability for centered difference discretization is $|\beta| \leq 1$ [65]. For these values of $\beta$ the matrix is irreducibly diagonally dominant [40]. Also, Elman & Golub showed in [40], that for these values of $\beta$, there exists a real diagonal matrix $Q$ such that $Q^{-1} A Q$ is symmetric positive definite. For solving a tridiagonal system, Gaussian elimination is clearly the preferred numerical technique, and from either one of the above two observations it follows that if $|\beta| \leq 1$ the algorithm is stable even without pivoting. If $|\beta| > 1$ the matrix $A$ is not diagonally dominant, and pivoting needs to be used to ensure
Chapter 1. Introduction

stability.

Alternatively, consider applying the point-Jacobi iterative scheme; the Jacobi iteration matrix is tridiagonal with zeros along its diagonal, and convergence results can be readily obtained by using the following (see, for example [40], or [95]):

**Lemma 1.1.** The eigenvalues of an \( n \times n \) tridiagonal matrix with \( b \), \( a \) and \( c \) along its subdiagonal, main diagonal, and superdiagonal respectively, are given by

\[
\lambda_i = a + \text{sign}(c)2\sqrt{bc}\cos\left(\frac{\pi i}{n+1}\right), \quad i = 1, \ldots, n .
\]  

(1.33)

By Lemma 1.1 the Jacobi scheme is convergent only if \( \beta \leq 2 \) [40]. Using the Jacobi scheme for solving a tridiagonal system is generally not efficient, but the fact that the scheme converges for such a narrow range of mesh Reynolds numbers is a primary indication of the difficulties that arise (and in fact are magnified) when considering two-dimensional or three-dimensional problems (where, as opposed to 1D problems, iterative methods are efficient).

We now compare the above with the analogous properties of the cyclically reduced matrix. The reduced matrix (for odd \( n \)), after scaling by 2, is [40]:

\[
S = \begin{pmatrix}
2 + 2\beta^2 & -(1 - \beta)^2 \\
-(1 + \beta)^2 & 2 + 2\beta^2 & -(1 - \beta)^2 \\
-(1 + \beta)^2 & 2 + 2\beta^2 & -(1 - \beta)^2 \\
& \ddots & \ddots & \ddots \\
& & & \ddots & \ddots & \ddots \\
& & & & \ddots & \ddots & \ddots \\
& & & & & \ddots & \ddots & \ddots \\
& & & & & & 2 + 2\beta^2 & -(1 - \beta)^2 \\
& & & & & & -(1 + \beta)^2 & 2 + 2\beta^2 \\
& & & & & & -(1 + \beta)^2 & 2 + 2\beta^2 \\
\end{pmatrix}
\]  

(1.34)

It can now be observed that the reduced matrix is diagonally dominant for all values of \( \beta \). In addition, it is symmetrizable by a real diagonal matrix for all \( \beta \) and the symmetrized matrix is positive definite [40]. Comparing that to the original system, we see that the restriction \( |\beta| \leq 1 \)
Chapter 1. Introduction

has disappeared. (We note, however, that the symmetrization operation itself, for both the reduced and unreduced matrix, may be numerically unstable [40]). As far as the point-Jacobi scheme is concerned, it has been shown in [40] that it is convergent for $\beta \neq 1$. In comparison to the condition of convergence for the unreduced solver ($\beta \leq 2$), this is a substantial improvement.

Additional observations regarding the eigenvalues and the spectral condition numbers of the unreduced and the reduced matrices are presented below:

**Proposition 1.1.** Suppose $|\beta| \leq 1$, $n$ is odd, and $h$ is sufficiently small. Denote the eigenvalues of the unreduced and the unscaled reduced matrices by $\lambda_j^{(U)}$ and $\lambda_j^{(R)}$ respectively. Then

$$\min_j \lambda_j^{(U)} \leq \min_j \lambda_j^{(R)} \leq \max_j \lambda_j^{(R)} \leq \max_j \lambda_j^{(U)} .$$  (1.35)

**Proof.** By Lemma 1.1, the eigenvalues of the unreduced matrix are:

$$\lambda_j^{(U)} = 2 - 2 \cdot \sqrt{1 - \beta^2 \cdot \cos(\pi j \bar{h})} , \quad j = 1, \ldots, n .$$  (1.36)

The eigenvalues of the unscaled reduced matrix (namely $\frac{1}{2}S$) are

$$\lambda_j^{(R)} = 1 + \beta^2 - (1 - \beta^2) \cdot \cos(\pi j \bar{h}) , \quad j = 1, \ldots, \frac{1}{2}(n-1) ,$$  (1.37)

where $\bar{h} = \frac{n-1}{2} + 1$. Since $\max_j \lambda_j^{(U)} = 2 + 2 \cdot \sqrt{1 - \beta^2 \cdot \cos(\pi \bar{h})}$ and $\max_j \lambda_j^{(R)} = 1 + \beta^2 + (1 - \beta^2) \cdot \cos(\pi \bar{h})$, and since $1 + \beta^2 \leq 2$, $1 - \beta^2 \leq 2 \sqrt{1 - \beta^2}$ and $\cos(\pi \bar{h}) \leq \cos(\pi h)$, it readily follows that $\max_j \lambda_j^{(R)} \leq \max_j \lambda_j^{(U)}$. As for the smallest eigenvalues, $\min_j \lambda_j^{(U)} = 2 - 2 \cdot \sqrt{1 - \beta^2 \cdot \cos(\pi \bar{h})}$ and $\min_j \lambda_j^{(R)} = 1 + \beta^2 - (1 - \beta^2) \cdot \cos(\pi \bar{h})$. Denote $x = \sqrt{1 - \beta^2}$. Then $\min_j \lambda_j^{(R)} - \min_j \lambda_j^{(U)} = 2x \cdot (1 - x) + o(h)$; this expression is nonnegative for $h$ sufficiently small. 

Fig. 1.5 illustrates Prop. 1.1 for a particular example: $n = 65$ and $\beta = 0.3$. Here the smallest and largest eigenvalues are 0.1841 and 1.9959 for the reduced matrix, and 0.0943 and 3.9057 for the unreduced matrix. The condition numbers are 72.58 and 262.27 - a ratio of about 3.6 in favor of the reduced matrix. Numerical experiments that we have performed, indicate
Chapter 1. Introduction

that for $|\beta| \leq 1$ the reduced matrix is better conditioned than the unreduced matrix by a factor of 2 to 4. In fact, for the symmetric case $\beta = 0$ it can be shown that:

**Proposition 1.2.** For $h$ sufficiently small, $\kappa_2(A) \approx 4\kappa_2(S)$. In other words, the reduced matrix is better conditioned, and the improvement is by a factor of approximately 4.

**Proof.** The two matrices in this case are identical (as far as the values on each diagonal are concerned), except the reduced matrix is about half the size of the unreduced matrix. By the eigenvalues specified in Prop. 1.1 it follows that $\kappa_2(A) = \frac{1+\cos(\pi\beta)}{1-\cos(\pi\beta)}$ and $\kappa_2(S) = \frac{1+\cos(\pi\beta)}{1-\cos(\pi\beta)}$. Using Taylor expansions, for $h$ sufficiently small $\kappa_2(A) = \frac{1}{\pi^2 h^2} + o(h^2)$ and $\kappa_2(S) = \frac{1}{\pi^2 \tilde{h}^2} + o(\tilde{h}^2)$, and since $\tilde{h} \approx 2h$, the result readily follows.

![Figure 1.5: eigenvalues of the reduced matrix (solid line) and the unreduced matrix (broken line) for the one-dimensional model problem with $n = 65$ and $\beta = 0.3$.](image)

We have seen, then, by examining a one-dimensional model problem, that one step of cyclic reduction improves the numerical properties of the linear system. In addition, since it involves solving a tridiagonal system for only half of the unknowns and a diagonal system for the rest of the unknowns, the cyclic reduction step gives rise to a faster solution procedure. In finite arithmetic and in situations of very large linear systems, the improvement can be very significant.
1.2.2 More Space Dimensions

In more than one space dimension the equations are considerably more complicated and the difficulties directly affect the level of difficulty in numerically solving the underlying discretized system of equations. From the linear algebra point of view, a major difference between the one-dimensional problem and higher dimensions, is that for the latter there is no way to turn the matrix into one with a dense band, regardless of the ordering strategy used. The natural lexicographic ordering for 2D and 3D is illustrated in Fig. 1.6; the corresponding sparsity patterns are depicted in Fig. 1.7. The red/black ordering in two dimensions is depicted in Fig. 1.8.

Another major difference is that more than one convective term is involved; thus, we can now face a situation of small and large Reynolds numbers simultaneously. When a discretization gives rise to a matrix which is not diagonally dominant, there might be a need for block methods which are performed in accordance with how weakly or strongly coupled the unknowns are in each of the directions [6]. Finite difference schemes and orderings that take into account the direction of the flow or attempt to minimize the truncation error might perform better than standard schemes [92],[98],[39]. In [92], Roe & Sidilkover derive multidimensional equivalents of the upwinding scheme in a direct fashion (that is, not by generalizing the 1D scheme in a dimension-by-dimension fashion). In particular, they examine linear schemes and determine the range of values of coefficients for which the truncation error is minimized over all positive schemes. The formulas which were derived depend on the geometry of the characteristics in the time-dependent problem. As is shown in [92], the cross-stream diffusion is smaller for the suggested scheme, compared to the dimensional upwind scheme, and these schemes have typically narrow stencils. See [99] for an overview of several narrow schemes with small truncation error for convection-dominated equations.

The procedure of one step of cyclic reduction can be performed for any two-dimensional and three-dimensional problem originally discretized by a finite difference scheme. When the five-point and seven-point operators are used, the difference equation for a red point depends
only on the point itself and its neighboring black points; similarly for the difference equations corresponding to the black points, and therefore the matrix $B$ is diagonal, and the cost of performing one step of cyclic reduction is low. However, the matrix $B$ in Eq. (1.31) is not always diagonal.

In higher dimensions cyclic reduction causes some loss of sparsity (as opposed to the 1D case). Another important difference from the 1D case is that the elimination must be accompanied by re-ordering of the unknowns (equivalently, permutation of the matrix). The re-ordering is essential because orderings which are effective for the standard tensor-product grid are not necessarily effective for the reduced grid, which has “holes” that correspond to the now-eliminated red points.

![Figure 1.6: natural lexicographic ordering of the tensor-product grids.](image)

In the early 1990s, Elman & Golub conducted analysis and an extensive set of numerical experiments, examining the properties of one-dimensional and two-dimensional non-self-adjoint problems to which one step of cyclic reduction is applied. Their findings were published in a series of papers [40],[41],[42]. They showed that in the two-dimensional case the cyclically reduced operator is a 9-point operator (different from the classical compact 9-point operator), and a matrix-vector product with the reduced matrix is slightly less costly than a matrix-vector product with the unreduced, original matrix. One-line and two-line orderings were considered, and bounds on convergence rates were derived for both orderings. Symmetrization results were
Chapter 1. Introduction

Figure 1.7: sparsity patterns of the matrices.

\[
\begin{array}{cccc}
15 & 7 & 16 & 8 \\
5 & 13 & 6 & 14 \\
11 & 3 & 12 & 4 \\
1 & 9 & 2 & 10 \\
\end{array}
\]

(a) ordering \hspace{2cm} (b) matrix

Figure 1.8: red/black ordering in the two-dimensional case.

given, which showed that the reduced matrix can be symmetrized by a real diagonal similarity transformation for a larger region of mesh Reynolds numbers, compared to the unreduced matrix; both matrices can be symmetrized for any mesh Reynolds number if upwind differences are used, or if centered differences are used and the mesh Reynolds numbers are smaller than 1 in magnitude. However, only reduced matrix can be symmetrized if both mesh Reynolds numbers are larger than 1 in magnitude and centered differences are used. The symmetrization was used to derive tight bounds on convergence rates of block stationary methods \cite{40,41}. Earlier observations of Thompson, Ferziger & Golub indicate convergence of the reduced SOR
Chapter 1. Introduction

solver for any mesh Reynolds number, and convergence which is faster than the convergence of unreduced solvers. Elman & Golub showed analytically as well as experimentally, that reduced solvers converge faster. In the constant coefficient case it was shown in [40] that the reduced matrix is a diagonally dominant $M$-matrix for any value of the PDE coefficients when upwind differencing is used, or for PDE coefficients in the diffusion-dominated region when centered differencing is used. The variable coefficient case was also addressed [42], and an extensive set of numerical experiments demonstrating performance and the connection between orderings and the direction of the flow, was given.

The work of Elman & Golub is very relevant to the current work. Even though the matrices derived and analyzed in this work are very different from the ones in [40],[41],[42], some of the same techniques are used. Throughout the thesis we will point out similarities and differences between the 3D case discussed here and the findings of Elman & Golub for the two-dimensional case.

The step of cyclic reduction could be repeated several times. The complete cyclic reduction approach involves repeating the decoupling until the reduced systems are small enough that they can be solved directly. This procedure was originally presented by Hockney [71], based on Golub’s ideas. Buneman [18] pointed out that computing the right-hand-side vector in finite precision arithmetic could result in severe round-off errors, and suggested a numerically stable algorithm (mathematically equivalent to Hockney’s algorithm). Buzbee, Golub and Nielson [20] presented Buneman’s algorithm and a variation of it, and provided careful analysis of the stability issue.

1.3 Thesis Outline and Notation

This work is the first to derive and analyze one step of cyclic reduction in conjunction with the important class of three-dimensional non-self-adjoint elliptic problems. In the succeeding chapters, we demonstrate the superiority of the reduced linear system of equations, in particular the convergence rate and amount of computational work involved.
Chapter 1. Introduction

In Chapter 2 we describe cyclic reduction and derive the cyclically reduced operator for the three-dimensional problem. In Chapter 3 we present block ordering strategies which can be applied to any three-dimensional problem, and demonstrate how they are applied to the reduced grid. Full details on the structures of the matrices associated with the various orderings are given. In Chapter 4 a comprehensive convergence analysis is performed for block stationary methods, and it is shown that the new system of equations is in general better conditioned than the original. Symmetrization conditions and tight bounds on convergence rates are derived. In Chapter 5 we discuss ways of solving the reduced system using Krylov subspace methods in conjunction with several preconditioning techniques. We give details on implementation, and present numerical experiments which include comparisons with the unreduced system. Finally, in Chapter 6 we draw conclusions and suggest future directions of investigation.

For notational convenience, the following rules are used throughout the thesis:

- For narrow banded matrices the terms 'diag', 'tri', 'penta' etc. are used in the same manner as in the Matlab command 'spdiags', that is: if \( x, y \) and \( z \) are vectors of size \( n \), then \( \text{tri}[x, y, z] \) will denote a tridiagonal matrix whose main diagonal consists of entries of the vector \( y \), the subdiagonal consists of \( x_1 \) to \( x_{n-1} \), and the superdiagonal consists of \( z_2 \) to \( z_n \). \( x_n \) and \( z_1 \) do not appear in the matrix in this case. \( x, y \) and \( z \) could be also matrices, in which case the same rules (as for vectors) apply.

- The index of a given diagonal in a matrix is: 0 for the main diagonal, positive numbers for superdiagonals and negative numbers for subdiagonals.

- \( I_n \) stands for the identity matrix of order \( n \).

- \( \rho(T) \) stands for the spectral radius of the matrix \( T \).

- If \( A \) and \( B \) are matrices, then \( B > A \) or \( B \geq A \) refer to inequalities elementwise.

- \( E_{s_1,\ldots,s_k} \) denotes a vector whose entries are \( s_1,\ldots,s_k \), repeated. For example \( E_{01} = (0,1,0,1,0,1,\ldots) \), \( E_{100} = (1,0,0,1,1,0,0,1,\ldots) \) and so on. The size of the vector will be clear from the context where it appears.
Chapter 2

Cyclic Reduction

In the Introduction the term "one step of reduction" was explained and its advantages in the one-dimensional and two-dimensional cases were briefly described. As was mentioned, several steps of cyclic reduction can be performed, until the system is small enough to be solved directly [71][18][20]. In Sec. 2.1 we present this procedure, namely complete cyclic reduction. The purposes of presenting it are: first, to clarify how cyclic reduction can be carried out for general block tridiagonal systems, and how the step can be iterated. Secondly, we mention the issues that arise when seeking to compute the solution accurately.

It was shown in [20] that complete cyclic reduction can be performed in a stable fashion if the matrix associated with the linear system arises from Poisson’s equation, i.e. it is symmetric positive definite. However, in the nonsymmetric case, where typically the matrix is not diagonally dominant, and does not necessarily have real eigenvalues, the stability of the complete cyclic reduction algorithm cannot be guaranteed. Therefore, in non-self-adjoint problems, the focus is on applying a single step of cyclic reduction. In Section 2.2 we derive the cyclically reduced operator for the three-dimensional case, and in Section 2.3 we present some of its properties.
Chapter 2. Cyclic Reduction

2.1 Complete Cyclic Reduction

Consider the linear system \( Ax = b \), where \( A \) is an \( n^2 \times n^2 \) block tridiagonal matrix of the form:

\[
A = \begin{pmatrix}
    B & C & 0 & 0 & \ldots & 0 \\
    C & B & C & 0 & \ldots & 0 \\
    0 & C & B & C & \ddots & \ddots \\
    \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\
    \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\
    C & B & C & 0 & \ldots & B \\
    0 & C & B & \ldots & \ldots & \ldots \\
\end{pmatrix}
\]  \hspace{1cm} (2.1)

and \( B \) and \( C \) are \( n \times n \) matrices. \(^1\) Suppose also that \( n = 2^k - 1 \), where \( k \) is some positive integer (see Sweet [105], [106] for an explanation on how to perform cyclic reduction if \( n \) is arbitrary).

In the following it is assumed that when the index exceeds the dimensions of the matrix the corresponding value is considered as being identically zero. For any even \( 2 \leq j \leq n - 1 \), we write three consecutive block equations of this system, as follows:

\[
\begin{align*}
C x_{j-2} + B x_{j-1} + C x_j &= b_{j-1} \\
C x_{j-1} + B x_j + C x_{j+1} &= b_j \\
C x_j + B x_{j+1} + C x_{j+2} &= b_{j+1}
\end{align*}
\]

If we now multiply the first and third block equations by \( C \), the second block equation by \(-B\), and add up the three block equations, we get:

\[
C^2 x_{j-2} + (2C^2 - B^2)x_j + C^2 x_{j+2} = C b_{j-1} + C b_{j+1} - B b_j . \hspace{1cm} (2.2)
\]

We thus obtain a new system of equations, for the vector \((x_2, x_4, \ldots, x_{n-1})^T\), where the corre-

\(^1\) In general there is no need to assume equality between the number of blocks and the size of each of the blocks. It is done here merely for the purpose of presentation.
Chapter 2. Cyclic Reduction

The corresponding matrix is

\[
\begin{pmatrix}
2C^2 - B^2 & C^2 & & & \\
C^2 & 2C^2 - B^2 & C^2 & & \\
0 & C^2 & 2C^2 - B^2 & C^2 & \\
& & \ddots & \ddots & \ddots \\
& & & C^2 & 2C^2 - B^2 \\
0 & C^2 & 2C^2 - B^2 & & \\
\end{pmatrix}
\]

and the right-hand-side is

\[
\begin{pmatrix}
C b_1 + C b_3 - B b_2 \\
C b_3 + C b_5 - B b_4 \\
\vdots \\
C b_{n-2} + C b_n - B b_{n-1}
\end{pmatrix}
\]

We can proceed recursively, as follows: define \( B^{(0)} = B \), \( C^{(0)} = C \), \( b_j^{(0)} = b_j \), and for \( m \geq 0 \),

\[
B^{(m+1)} = 2(C^{(m)})^2 - (B^{(m)})^2; \quad C^{(m+1)} = (C^{(m)})^2, \quad b_j^{(m+1)} = C^{(m)}(b_{j-2m}^{(m)} + b_{j+2m}^{(m)}) - B^{(m)}b_j^{(m)}.
\]

We now have a system of equations of the form \( A^{(m)}x^{(m)} = c^{(m)} \), where

\[
A^{(m)} = 
\begin{pmatrix}
B^{(m)} & C^{(m)} \\
C^{(m)} & B^{(m)} & C^{(m)} \\
0 & C^{(m)} & B^{(m)} & C^{(m)} \\
& & \ddots & \ddots & \ddots \\
& & & C^{(m)} & B^{(m)} & C^{(m)} \\
0 & C^{(m)} & B^{(m)} & & & \\
\end{pmatrix}
\]

and the elements of \( c^{(m)} \) are given by \( c_j^{(m)} = b_j^{(m)} \). Once this system is solved, it is straightforward to compute the solution for the unknowns that were previously eliminated throughout the process.
Chapter 2. Cyclic Reduction

Note that even if $B$ and $C$ are sparse, $B^{(m)}$ and $C^{(m)}$ are not necessarily sparse. For solving the reduced system, Fourier type algorithms based on diagonalizing the matrix (by the matrix whose columns are the eigenvectors) could be used. This option is very attractive when $B$ and $C$ commute ($BC = CB$), as in this case these two matrices can be diagonalized simultaneously. Since $B^{(m)}$ and $C^{(m)}$ are polynomials in $B$ and $C$, it follows that their eigenvalues can be easily computed, provided that the eigenvalues of $B$ and $C$ are known. Indeed, for Poisson’s equation the spectrum of the matrix is known, and stability analysis is available [20].

The problematic term in the computation of the right-hand-side is $B^{(m)}b^{(m)}_j$. There is more than one way to employ a recurrence relation for computing it. For example [20]: $v_0 = -2b^{(m)}_j; v_1 = Bb^{(m)}_j; v_k = -Bv_{k-1} - C^2v_{k-2}$. Using this formula, we get $B^{(m)}b^{(m)}_j = v_2m$.

Suppose $\lambda_j$ and $\omega_j$ are the eigenvalues of $B$ and $C$ respectively. It is shown in [20] that if $|\lambda_j| < 2|\omega_j|$ then the computation of $B^{(m)}b^{(m)}_j$ is actually done by a recurrence relation involving the term $\cosh(jz_j)$, where $z_j = \cosh^{-1}\left(\frac{-\lambda_j}{2\omega_j}\right)$. Thus there can be a significant difference in magnitude between $b^{(m)}_1$ and $b^{(m)}_n$, and the latter could be lost in rounding errors if $n$ is large. (This situation occurs frequently; for example, it occurs in the finite difference discretization of Poisson’s equation).

Buneman’s algorithm [18] for overcoming the difficulty involves “backward”-type computations. For Poisson’s equation, where $C$ is the identity matrix, the technique uses the equality $B = B^{(1)}B^{-1} - 2B^{-1}$; and in general $B^{(m)} = 2I_m - B^{(m+1)}$. Substituting this for $j = 1, \ldots, m$, after some algebraic manipulation, a two-step recurrence relation is obtained, where the term $(B^{(m-1)}B^{(m-2)} \cdots B^{(0)})^{-1}$ replaces the problematic term in the original algorithm. It is then shown that the 2-norm of the matrix $(B^{(m-1)}B^{(m-2)} \cdots B^{(0)})^{-1}$ is bounded from below by $e^{-c\theta_1}$, where $c$ is some constant, and $\theta_1 = \cosh^{-1}(-\lambda_i/2)$. For the Poisson’s equation, $\theta_1 > 1$, and thus the upper bound is small enough to ensure stability. The full proof of stability of this procedure can be found in [20, pp. 648-655].

Throughout the years, many extensions to the original papers and algorithms [71],[18],[20] have been presented. Buzbee, Dorr, George & Golub [19] use cyclic reduction for solving Pois-
Cyclic Reduction

son’s equation on irregular regions; Concus & Golub [25] discuss two-dimensional nonseparable cases; Bauer and Reiss [12] generalize the procedure for block pentadiagonal systems arising from the discrete biharmonic operators; Heller [68] compares the iteration count of block cyclic reduction to that of block Gaussian elimination [111] and shows that when the matrix is diagonally dominant, norms of the off-diagonal blocks decrease quadratically relative to the diagonal blocks at each step of reduction, and based on this observation, suggests criteria for early termination of the process, when the matrix becomes essentially block diagonal. Detyna [33] suggests an algorithm which is based on introducing two sets of equations, each with a different stencil, whose solutions are identical, and using the deliberately created large number of degrees of freedom, to eliminate half of the equations in each of these two sets, in a manner that preserves the structure of the computational molecules (of each of the sets). This step can be repeated, and leads to a procedure with \( O(N^2) \) operations, thus is a faster algorithm than the classical algorithm [71],[18],[20] described in detail above; on the other hand, the algorithm gives rise to larger errors in the numerical solution; the error can be reduced by applying the suggested algorithm on the finer grids, and proceeding at the higher grids in the hierarchy with the more stable cyclic reduction algorithm (or any other accurate fast solver). Bondeli & Gander discuss application of cyclic reduction for special tridiagonal systems [15]; Swarztrauber & Sweet [104], Gallopoulos & Saad [51], Amodio & Mastronardi [2] discuss aspects of parallelization; Amodio & Paprzycki [3] present a parallel algorithm for solving boundary value problems using cyclic reduction, and apply it to a distributed memory machine. A description of the cyclic reduction idea and a list of references can be found in Golub & Van Loan [55].

2.2 The Three-Dimensional Cyclically Reduced Operator

Even though the complete cyclic reduction algorithm can be carried out even when \( B \) and \( C \) do not commute [20], for nonsymmetric matrices the spectrum is frequently not known and numerical stability cannot be ensured in general. For example, Heller’s observations in [68] are not applicable, as no diagonal dominance is guaranteed. In three dimensions issues of fill-in and stability are magnified, and could be very problematic. It is our purpose, then, to focus in
Chapter 2. Cyclic Reduction

a single step of cyclic reduction and perform comprehensive analysis.

As a first step we derive the cyclically reduced operator. The derivation is done by performing the block Gaussian elimination step described in the Introduction [see Eq. (1.31) and (1.32)]. Once the points corresponding to one of the colors (say red) are eliminated, the resulting reduced grid has a somewhat irregular structure: there are "holes" in the grid, which correspond to the previously present red points. A difficulty which is specific to three dimensions is: the parity of the planes has to be taken into consideration; points of the same color are not located on the same spots for even-indexed planes and odd-indexed planes. This can be illustrated by looking at the three-dimensional checkerboard depicted in Fig. 2.1.

![Figure 2.1: a three-dimensional checkerboard.](image)

Our starting point is the original, unreduced problem in three dimensions. The red/black ordering is depicted in Fig. 2.2; the points that have to do with the block elimination are numbered in Fig. 2.3. Here the point for which the discretization is done is in the center, indexed by #13. If this point is black, then points #4, #9, #12, #14, #17 and #22 are red and are to be eliminated. The other points in the figure are black, but their corresponding entries in the matrix at row #13 are to be changed after eliminating the red points. The construction of the reduced computational molecule is done by eliminating half of the rows and the columns of the matrix that corresponds to red/black ordering. The constant coefficient
Figure 2.2: red/black ordering, and the corresponding sparsity pattern of the associated matrix, for the three-dimensional problem.

case is easier to derive, as the computational molecule is independent of the coordinates of the gridpoints. We thus illustrate how to construct the cyclically reduced operator for this case first.

2.2.1 The Constant Coefficient Model Problem

Consider the model problem (1.15). In terms of the matrix elements, below are the entries which are affected by the block elimination step for the model problem.

Here are the computations performed during the block elimination:
From this, we can see that the typical value on the diagonal of the reduced matrix is:

$$a^{-1}(a^2 - 2bc - 2cd - 2fg). \quad (2.6)$$

By 'typical' we mean an entry that is associated with an interior point. For non-interior points fewer operations of elimination are required, and the value in their associated diagonal entry changes with respect to (2.6) in the following manner:

- **add** $a^{-1}cd$ in case the $x$-coordinate of the associated point $(ih, jh, kh)$ is $i = 1$ or $i = n$.
- **add** $a^{-1}be$ in case the $y$-coordinate of the associated point $(ih, jh, kh)$ is $j = 1$ or $j = n$.
- **add** $a^{-1}fg$ in case the $z$-coordinate of the associated point $(ih, jh, kh)$ is $k = 1$ or $k = n$. 
Let $R$ be the reduced operator for the model problem, after scaling by $ah^2$. Then for an interior gridpoint, $(ih, jh, kh)$, the difference operator is given by:

$$R u_{i,j,k} = (a^2 - 2be - 2cd - 2fg) u_{i,j,k} - f^2 u_{i,j,k-2} - 2ef u_{i,j+1,k-1}$$

$$-2ef u_{i-1,j,k-1} - 2df u_{i+1,j,k-1} - 2bf u_{i,j-1,k-1} - e^2 u_{i,j+2,k}$$

$$-2de u_{i+1,j+1,k} - c^2 u_{i-2,j,k} - d^2 u_{i+2,j,k} - 2bc u_{i-1,j-1,k}$$

$$-b^2 u_{i,j-2,k} - 2eg u_{i,j+1,k+1} - 2cg u_{i-1,j,k+1} - 2ce u_{i-1,j+1,k}$$

$$-2bd u_{i+1,j-1,k} - 2dg u_{i+1,j,k+1} - 2bg u_{i,j-1,k+1} - g^2 u_{i,j,k+2}.$$  \hfill (2.7)

Figure 2.4: structure of the computational molecule associated with the reduced operator. The gray squares correspond to the gridpoints which have been eliminated throughout the elimination process.

Referring to the indices used in Figure 2.3, in terms of mesh Reynolds numbers the values of the computational molecule (for an interior gridpoint) are given below.
If the points in the reduced grid are ordered lexicographically, then the sparsity pattern of the matrix is as in Fig. 2.5.

Notice that the matrix corresponds to a grid of size $6 \times 6 \times 6$, but is only $108 \times 108$ in size, whereas $6^3 = 216$. This is because half of the unknowns in the original linear system were eliminated. Another thing to notice is that the matrix is less sparse than the original,
unreduced matrix: there are typically 19 nonzeros in each rows, as opposed to (typically) 7 nonzeros in each row of the unreduced matrix. Finally, notice that the main block diagonal has "holes" within itself, which get "larger" when the size of the matrix is larger. This is because no re-ordering which better fits the structure of the reduced grid has been performed here. This aspect is to be addressed in Chapter 3.

In the discussion that follows, we refer to centered schemes. Similar observations can be made for upwind schemes. Denote the continuous operator corresponding to the model problem by

$$L = -\Delta + (\sigma, \tau, \mu)^T \nabla.$$  \hspace{1cm} (2.8)

Expanding (2.7) in a multivariate Taylor expansion about the gridpoint \((ih, jh, kh)\) yields:

$$2Ru = Lu - \frac{1}{6} h^2 u_{yyyy} - \frac{1}{6} h^2 u_{yyzz} - \frac{1}{6} h^2 \tau \mu u_{yz} + \frac{1}{6} h^2 \mu u_{yyz} + \frac{1}{6} h^2 \sigma u_{xxx}$$

$$+ \frac{1}{6} h^2 \tau u_{xxy} + \frac{1}{6} h^2 \sigma u_{xyy} + \frac{1}{3} h^2 \sigma u_{xzy} - \frac{1}{6} h^2 \sigma \mu u_{xxz} - \frac{1}{6} h^2 \tau \mu u_{xyy} + \frac{1}{3} h^2 \mu u_{zzz}$$

$$- \frac{1}{12} h^2 \mu^2 u_{xxz} + \frac{1}{12} h^2 \tau u_{xyy} - \frac{1}{12} h^2 \tau^2 u_{yy} + \frac{1}{6} h^2 \tau u_{yz} - \frac{1}{6} h^2 u_{xxyz} - \frac{1}{6} h^2 u_{xxzz}$$

$$+ \frac{1}{6} h^2 \mu u_{xxy} - \frac{1}{12} h^2 \tau^2 u_{xx} - \frac{1}{6} h^2 u_{xxzz} - \frac{1}{6} h^2 u_{zzzz} + O(h^3).$$  \hspace{1cm} (2.9)

The above computation was carried out using Maple V. The right-hand-side in Eq. (2.9) can be referred to as an introduction of a new continuous operator, based on a combination
of the original equation, and $O(h^2)$ terms associated with second or higher derivatives of the solution. This continuous operator can be discretized directly, using the points that belong to the computational molecule of the cyclically reduced operator. For example,

$$u_{xx} \approx \frac{u_{i+2,j,k} - 2u_{i,j,k} + u_{i-2,j,k}}{h^2},$$

and similarly for $u_{yy}$ and $u_{zz}$; and for the first derivatives

$$u_x \approx \frac{u_{i+1,j+1,k} + u_{i+1,j-1,k} - u_{i-1,j+1,k} - u_{i-1,j-1,k}}{4h},$$

and similarly for $u_y$ and $u_z$. The result is a discretization scheme with an $O(h^2)$ discretization error. This result is similar to the result in [57] for the two-dimensional case.

We also state that the reduced right-hand-side is equal to $w_{i,j,k}$ with an $O(h^2)$ error. Indeed, Gaussian elimination yields the following right hand side:

$$w_{i,j,k} - \frac{b}{a} w_{i,j-1,k} - \frac{c}{a} w_{i-1,j,k} - \frac{d}{a} w_{i+1,j,k} - \frac{e}{a} w_{i,j+1,k} - \frac{f}{a} w_{i,j,k-1} - \frac{g}{a} w_{i,j,k+1},$$

whose Taylor expansion about the gridpoint $(ih, jh, kh)$, after scaling by $2ah^2$ [that is, after doing the same scaling as for Eq. (2.9)] is: $w - \frac{h^2}{12} [-\Delta w + (\sigma w_x + \tau w_y + \mu w_z) ]$, where all the derivatives of $w$ are evaluated at the point $(ih, jh, kh)$. This result is similar to the result in [41] for the two-dimensional case.

### 2.2.2 The Variable Coefficient Case

For the variable coefficient case performing cyclic reduction is considerably more difficult, as now the components of the computational molecule depend on each gridpoint’s coordinates. We first illustrate how the seven-point operator is derived. This derivation is standard; see for example [41] for details on the two-dimensional case. The discrete difference operator is derived by working on a fine grid whose size is $\frac{h}{2}$ rather than $h$.

Equation (1.1) is rewritten in the following manner:

$$-\left[(p u_x)_x + (q u_y)_y + (r u_z)_z\right] + s u_x + t u_y + v u_z = w.$$  \hspace{1cm} (2.11)

Assume that $p, q, r > 0$ on $\Omega$. The seven-point discretization is done as follows:
Chapter 2. Cyclic Reduction

- For the convective terms a centered finite difference approximation or an upwind discretization scheme is used. For example, if \( s > 0 \) on \( \Omega \), the term involving the derivative in the \( x \)-direction is discretized either as follows (centered differencing): 
  \[
  s_{u_x} \approx \frac{s_{i,j,k} u_{i+1,j,k} - s_{i,j,k} u_{i-1,j,k}}{2h},
  \]
  or as follows (upwind backward differencing):
  \[
  s_{u_x} \approx \frac{s_{i,j,k} u_{i+1,j,k} - s_{i,j,k} u_{i-1,j,k}}{h}.
  \]

The terms \( tu_y \) and \( vu_z \) are discretized in a similar fashion.

- For the diffusive terms a centered difference scheme is applied: a mesh of size \( \frac{h}{2} \) is used and the terms \( pu_x \), \( qu_y \) and \( ru_z \) are discretized, and then the derivatives of these terms in the \( x \), \( y \) and \( z \) directions respectively are computed. In doing so, a scheme that defines \( u \) at a grid whose mesh size is \( h \) is obtained, using values of the coefficient functions at the finer grid whose size is \( \frac{h}{2} \). For example, for the first term in Eq. (1.1)
  \[
  (pu_x)_x \approx \frac{p_{i+\frac{1}{2},j,k} u_{i+1,j,k} - (p_{i+\frac{1}{2},j,k} + p_{i-\frac{1}{2},j,k}) u_{i,j,k} + p_{i-\frac{1}{2},j,k} u_{i-1,j,k}}{h^2}.
  \]

The terms \((qu_y)_y\) and \((ru_z)_z\) are discretized by applying a similar procedure.

The difference operator for an interior gridpoint is given by:

\[
F_{u_{i,j,k}} = a_{i,j,k} u_{i,j,k} + b_{i,j,k} u_{i,j-1,k} + c_{i,j,k} u_{i-1,j,k} + d_{i,j,k} u_{i+1,j,k} + e_{i,j,k} u_{i,j+1,k} + f_{i,j,k} u_{i,j,k-1} + g_{i,j,k} u_{i,j,k+1},
\]

If \( s_{i,j,k}, t_{i,j,k} \) and \( v_{i,j,k} \) are all positive and centered differences are used, the values of the computational molecule are given by

\[
\begin{align*}
a_{i,j,k} &= p_{i+\frac{1}{2},j,k} + p_{i-\frac{1}{2},j,k} + q_{i,j+\frac{1}{2},k} + q_{i,j-\frac{1}{2},k} + r_{i,j,k+\frac{1}{2}} + r_{i,j,k-\frac{1}{2}}, \\
b_{i,j,k} &= -q_{i,j-\frac{1}{2},k} - \frac{t_{i,j,k} h}{2}; \quad c_{i,j,k} = -q_{i,j+\frac{1}{2},k} + \frac{t_{i,j,k} h}{2}; \\
c_{i,j,k} &= -p_{i-\frac{1}{2},j,k} - \frac{s_{i,j,k} h}{2}; \quad d_{i,j,k} = -p_{i+\frac{1}{2},j,k} + \frac{s_{i,j,k} h}{2}; \\
f_{i,j,k} &= -r_{i,j-\frac{1}{2},k} - \frac{v_{i,j,k} h}{2}; \quad g_{i,j,k} = -r_{i,j+\frac{1}{2},k} + \frac{v_{i,j,k} h}{2}.
\end{align*}
\]

If one uses upwind schemes, then the type of scheme depends on the sign of the convective terms. Assuming that \( s \), \( t \) and \( v \) do not change sign in the domain, for each of them which is
positive one would use the backward scheme, and forward scheme would be used in the negative case. Discretizing using backward differences yields

\[
\begin{align*}
a_{i,j,k} &= p_{i,j,k} + p_{i,j,k} + q_{i,j,k} + q_{i,j,k} + r_{i,j,k} + r_{i,j,k} + s_{i,j,k} \\
+ t_i j k h + v_i j k h ; b_{i,j,k} &= -q_{i,j,k} - t_i j k h ; \quad \epsilon_{i,j,k} = -q_{i,j,k} + \frac{1}{2} k ; \\
c_{i,j,k} &= -p_{i,j,k} - s_{i,j,k} - d_{i,j,k} ; \quad d_{i,j,k} = -p_{i,j,k} + \frac{1}{4} k ; \\
f_{i,j,k} &= -r_{i,j,k} - \frac{1}{2} - w_{i,j,k} h ; \quad g_{i,j,k} = -r_{i,j,k} + \frac{1}{4} h ,
\end{align*}
\]

and for forward differences the above needs to be modified in an obvious manner.

As opposed to the constant coefficient model problem [Eq. (1.20c)], now the components of the computational molecule do depend on the associated coordinates of the gridpoints. Nevertheless, the sparsity structure of the reduced matrix is identical in both cases. The entries of the matrix which are affected by the elimination are specified below.

\[
\begin{array}{cccccccc}
\text{column:} & (i,j,k-1) & (i,j+1,k) & (i-1,j,k) & (i+1,j,k) & (i,j-1,k) & (i,j,k+1) & (i,j,k) \\
\text{row:} (i,j,k-1) & a_{i,j,k} & & & & & & \\
\text{row:} (i,j+1,k) & & a_{i,j+1,k} & & & & & \\
\text{row:} (i-1,j,k) & & & a_{i-1,j,k} & & & & \\
\text{row:} (i+1,j,k) & & & & a_{i+1,j,k} & & & \\
\text{row:} (i,j-1,k) & & & & & a_{i,j-1,k} & & \\
\text{row:} (i,j,k+1) & & & & & & a_{i,j,k+1} & \\
\text{row:} (i,j,k) & & & & & & & a_{i,j,k} \\
\end{array}
\]

\[
\begin{array}{cccccccc}
\text{column:} & (i,j,k-2) & (i,j+1,k-1) & (i-1,j,k-1) & (i+1,j,k-1) & (i,j-1,k-1) & (i,j,k+2) & (i,j,k) \\
\text{row:} (i,j,k-1) & f_{i,j,k} & & e_{i,j,k} & & & & \\
\text{row:} (i,j+1,k) & & f_{i,j+1,k} & & & & & \\
\text{row:} (i-1,j,k) & & & f_{i-1,j,k} & & & & \\
\text{row:} (i+1,j,k) & & & & f_{i+1,j,k} & & & \\
\text{row:} (i,j-1,k) & & & & & f_{i,j-1,k} & & \\
\text{row:} (i,j,k+1) & & & & & & f_{i,j+1,k} & \\
\text{row:} (i,j,k) & & & & & & & f_{i,j,k} \\
\end{array}
\]

\[
\begin{array}{cccccccc}
\text{column:} & (i+1,j,k-1) & (i+1,j,k+1) & (i-2,j,k) & (i+2,j,k) & (i-1,j-1,k) & (i+1,j+1,k) \\
\text{row:} (i,j,k-1) & & & & & & & \\
\text{row:} (i,j+1,k) & & & & & & & \\
\text{row:} (i-1,j,k) & & & & & & & \\
\text{row:} (i+1,j,k) & & & & & & & \\
\text{row:} (i,j-1,k) & & & & & & & \\
\text{row:} (i,j,k+1) & & & & & & & \\
\text{row:} (i,j,k) & & & & & & & \\
\end{array}
\]
Let \( \tilde{R} \) denote the reduced discrete operator, after scaling by \( h^2 \). For an interior gridpoint, \((ih, jh, kh)\), the operation of \( \tilde{R} \) is given by the difference equation

\[
\tilde{R} u_{i,j,k} = (a_{i,j,k} - \frac{f_{i,j,k} g_{i,j,k-1}}{a_{i,j,k-1}} - \frac{e_{i,j,k} f_{i,j,k-1}}{a_{i,j,k-1}} - \frac{e_{i,j,k} h_{i,j,k+1}}{a_{i,j,k+1}} ) u_{i,j,k} \\
- \frac{d_{i,j,k} e_{i,j,k}}{a_{i,j,k}} u_{i+1,j,k-1} - \frac{e_{i,j,k} f_{i,j,k}}{a_{i,j,k}} u_{i-1,j,k-1} - \frac{e_{i,j,k} h_{i,j,k}}{a_{i,j,k}} u_{i-1,j,k+1} \\
- \frac{d_{i,j,k} e_{i,j,k}}{a_{i,j,k}} u_{i+1,j,k+1} - \frac{e_{i,j,k} f_{i,j,k}}{a_{i,j,k}} u_{i-1,j,k+1} - \frac{e_{i,j,k} h_{i,j,k}}{a_{i,j,k}} u_{i-1,j,k-1} \\
- \frac{d_{i,j,k} e_{i,j,k}}{a_{i,j,k}} u_{i+1,j,k+1} - \frac{e_{i,j,k} f_{i,j,k}}{a_{i,j,k}} u_{i-1,j,k+1} - \frac{e_{i,j,k} h_{i,j,k}}{a_{i,j,k}} u_{i-1,j,k-1} \\
- \frac{d_{i,j,k} e_{i,j,k}}{a_{i,j,k}} u_{i+1,j,k+1} - \frac{e_{i,j,k} f_{i,j,k}}{a_{i,j,k}} u_{i-1,j,k+1} - \frac{e_{i,j,k} h_{i,j,k}}{a_{i,j,k}} u_{i-1,j,k-1}.
\] (2.15)

Notice that Eq. (2.15) involves significantly more floating point operations compared to the seven-point operator. The point of computational work involved in constructing the reduced matrix is to be discussed in Chap. 5.

### 2.3 Properties of the Reduced Matrix

The reduced matrix \( S = E - DB^{-1}C \) is the *Schur complement* matrix of the original matrix \( A \). Schur complements arise in many applications. In particular, we mention the class of *domain decomposition* techniques [22]. These are techniques which are suitable for irregular (e.g. L-shaped) domains: the domain is divided into "simple" (e.g. rectangular) subdomains, where the
Chapter 2. Cyclic Reduction

partial differential equation can be easily solved. The difficulty is that for some partitionings, between each two subdomains there is an “interface” layer, of unknowns that belong to both subdomains (see e.g. [95]). The submatrix associated with the interface unknowns is the Schur complement and is typically a much smaller matrix than the submatrices associated with unknowns that belong to the subdomains. If the solution for the interface unknowns is obtained first, by solving for the Schur complement, then it is easy to find a solution for the rest of the unknowns. This approach is used in several areas of applications. For example, in cut-loop techniques for multibody systems simulation (see [7] and references therein).

Schur complements have many useful properties (see, for example, [27]). In the context of cyclic reduction, we are interested in seeing whether numerical properties of the unreduced matrix are preserved, or improved, after one step of cyclic reduction is performed. The results in Lemmas 2.1-2.4 below are not specific to the particular class of problems discussed in this thesis and are well known (see, for example, [95], for some of the results presented below); the discussion and results following after these lemmas are specific to our reduced system.

**Lemma 2.1.** If the original matrix $A$ is nonsingular, then the Schur complement matrix $S$ is nonsingular as well.

**Proof.** The block-LU decomposition of $A$ is

$$A = \begin{pmatrix} B & C \\ D & E \end{pmatrix} = \begin{pmatrix} I & 0 \\ DB^{-1} & I \end{pmatrix} \begin{pmatrix} B & C \\ 0 & S \end{pmatrix}$$

(2.16)

thus $S$ must be nonsingular. $\square$

**Lemma 2.2.** If the matrix $A$ is symmetric positive definite, then so is $S$.

**Proof.** It is straightforward to show that

$$\begin{pmatrix} B & 0 \\ 0 & S \end{pmatrix} = \begin{pmatrix} I & 0 \\ -C^T B^{-1} & I \end{pmatrix} \begin{pmatrix} B & C \\ C^T & E \end{pmatrix} \begin{pmatrix} I & -B^{-1}C \\ 0 & I \end{pmatrix}$$

(2.17)

and thus, if $A$ is symmetric positive definite, so is $S$. $\square$
Chapter 2. Cyclic Reduction

For $M$-matrices (see Defn. 1.3), we have the following result (the proof can be found in [45] or [9, Thm. 6.10]):

**Lemma 2.3.** If a matrix is an $M$-matrix, then so is the associated Schur complement matrix.

As far as conditioning is concerned, we have:

**Lemma 2.4.** If $A$ is symmetric positive definite, then

$$
\kappa_2(S) \leq \kappa_2(A) .
$$

(2.18)

In other words, the matrix $S$ is better conditioned than $A$.

**Proof.** From the proof of Lemma 2.2 $B$ is symmetric positive definite. Hence so are $B^{-1}$ and $C^T B^{-1} C$. Since $S = E - C^T B^{-1} C$, $E$ must be symmetric positive definite, and thus

$$
\|S\|_2 \leq \|E\|_2 \leq \|A\|_2 .
$$

(2.19)

On the other hand,

$$
A^{-1} = \begin{pmatrix}
B^{-1} + B^{-1} C S^{-1} D B^{-1} & -B^{-1} C S^{-1} \\
-S^{-1} D B^{-1} & S^{-1}
\end{pmatrix}
$$

(2.20)

so we have

$$
\|S^{-1}\|_2 \leq \|A^{-1}\|_2 .
$$

(2.21)

From Eq. (2.19) and (2.21) the result stated in the lemma readily follows.

Moving to the nonsymmetric case, we first note that, as already mentioned in the Introduction, the first step of cyclic reduction is attractive mainly because the matrix to be inverted is diagonal. In fact, matrices that arise from discretization of differential equations, using three-point, five-point or seven-point operators, all belong to a special class of matrices, defined as follows [67, §9.2]:
Chapter 2. Cyclic Reduction

Definition 2.1. Let $T$ be partitioned into $q^2$ submatrices:

$$
T = \begin{pmatrix}
T_{1,1} & \cdots & T_{1,q} \\
\vdots & \ddots & \vdots \\
T_{q,1} & \cdots & T_{q,q}
\end{pmatrix}
$$

(2.22)

$T$ is said to have block Property A with respect to this partitioning if there exist two disjoint nonempty subsets $S_R$ and $S_B$ of $\{1, 2, \ldots, q\}$ such that $S_R \cup S_B = \{1, \ldots, q\}$ and such that if $T_{i,j} \neq 0$ and $i \neq j$, then $i \in S_R$ and $j \in S_B$ or $i \in S_B$ and $j \in S_R$.

Property A was first defined by Young [116], and was later generalized by Arms, Gates & Zondek [4], and by Varga [112]. The matrices that are associated with the three-point, five-point and seven-point operators, all have Property A with respect to $1 \times 1$ matrices; showing it is easy: the two subsets $S_R$ and $S_B$ mentioned in Definition 2.1 correspond to the set of red gridpoints and the set of black gridpoints when red/black ordering is used. A matrix that corresponds to any arbitrary ordering of the unknowns can be transformed by a symmetric permutation to the matrix corresponding to red/black ordering, and thus it has Property A. The question whether a matrix has Property A or not, is very significant in the convergence analysis of stationary methods, in particular the SOR scheme [117]. We will discuss this in Chapter 4, where we will also address the question in what circumstances the reduced matrix has Property A, and relative to which partitionings.

In Fig. 2.6 plots and histograms of the eigenvalues of both the unreduced and the reduced systems are depicted, for the symmetric positive definite case corresponding to Poisson’s equation

$$
-\Delta u = f,
$$

(2.23)
discretized on a $12 \times 12 \times 12$ grid. In the histograms, on the $x$-axis we have the eigenvalues, and the $y$-axis is their number. The eigenvalues of the unreduced system are explicitly known
in analytical terms and are distributed between 0 and 12. The maximal eigenvalue is close to
12, and the minimal eigenvalue is close to 0. On the other hand, the reduced matrix has all
its eigenvalues between 0 and 6, and it is evident either by the histogram or by the moderate
slope of the graph for the eigenvalues (at the region of the largest ones), that the majority of
the eigenvalues are between 5 and 6. Also, the smallest eigenvalues of the reduced matrix are
more isolated compared to the unreduced matrix, and are larger than the smallest eigenvalues
of the unreduced matrix. In this particular case the minimal eigenvalues of the unreduced and
reduced matrices are 0.174 and 0.344 respectively. The condition number of the unreduced
matrix is larger by a factor of 3.885.

Since the rate of convergence of Krylov space methods strongly depends on the condition
number of the matrix and on clustering of the eigenvalues, the above observations lead to the
conclusion that convergence of the reduced solver will be faster.

![Graphs](image)

(a) unreduced  
(b) reduced

Figure 2.6: eigenvalues of both systems for Poisson’s equation on a $12 \times 12 \times 12$ grid

Similar observations apply to the general nonsymmetric case; when convection is moderate
in size, the structure of eigenvalues is very similar to the structure presented in Fig. 2.6. When
convection dominates, the situation is a bit different, but still, the reduced matrix is better conditioned.

In Fig. 2.7 we refer to a more difficult problem: the singularly perturbed equation

\[-\varepsilon \Delta u + \nabla v \cdot \nabla u = f,\]

where \(v = \frac{1}{2}[(x - \frac{1}{2})^2 + (y - \frac{1}{2})^2 + (z - \frac{1}{2})^2]\) and the domain is the unit cube \((0,1) \times (0,1) \times (0,1)\).

Upwind differencing is used to discretize the problem on a uniform grid. Taking \(\varepsilon = 0.02\) on an \(8 \times 8 \times 8\) grid, we get that the spectral condition number of the unreduced matrix is \(\kappa_2 = 2,292\) with \(\sigma_{\text{min}} = 0.0098\); \(\sigma_{\text{max}} = 22.46\) and the condition number of the reduced matrix is \(\kappa_2(R) = 759\), with \(\sigma_{\text{min}} = 0.0195\); \(\sigma_{\text{max}} = 14.80\). In Fig. 2.7 we give the histogram of the singular values of the matrix for this particular problem.

![Histogram of singular values](image)

Figure 2.7: singular values of both matrices for Eq. (2.24) \((8 \times 8 \times 8\) grid)

Another point of interest, is the circumstances in which the reduced matrix is an irreducibly diagonally dominant \(M\)-matrix. Here, in a way which is similar to [40, Cor. 3] we can obtain the following useful result:

**Lemma 2.5.** If \(be, cd, fg > 0\) then both the unreduced and the reduced matrices are diagonally dominant \(M\)-matrices.

**Proof.** Consider the unreduced matrix. For the matrix to be an \(M\)-matrix, a necessary condition is that the off-diagonal elements must be nonpositive, which is readily obtained if \(be, cd, fg > 0\). It is straightforward to see, by substituting the values of the computational molecule in (1.17),
Chapter 2. Cyclic Reduction

(1.18) or (1.19), that the unreduced matrix is diagonally dominant and irreducible. Thus by Theorem 1.1 the unreduced matrix is an $M$-matrix.

As for the reduced matrix, consider a row associated with an interior point: diagonal dominance is translated to the requirement that

$$a^2 - 2be - 2cd - 2fg \geq b^2 + c^2 + d^2 + e^2 + f^2 + g^2 + 2|bf| + 2|cf| + 2|df| + 2|ef|$$
$$+ 2|bc| + 2|bd| + 2|ce| + 2|de| + 2|bg| + 2|cg| + 2|dg| + 2|eg|.$$  \hspace{1cm} (2.25)

If (2.25) holds, then diagonal dominance of rows which correspond to points close to the boundary also holds, as in this case the associated diagonal entry is larger. Eq. (2.25) is identical to $a^2 \geq (|b| + |c| + |d| + |e| + |f| + |g|)^2$, which holds because the unreduced matrix is diagonally dominant. by Lemma 2.3 the reduced matrix is also an $M$-matrix. \hfill \Box

The results mentioned in this section are important in the sense that if the original matrix has valuable numerical properties, then so does the reduced matrix; the procedure of cyclic reduction does not damage properties such as diagonal dominance or positive definiteness.
Chapter 3
Ordering Strategies

The question of ordering of the unknowns is of importance for both iterative and direct solvers, as a good ordering strategy can lead to significant saving in computational work. Among books that address this subject we mention George & Liu [52] and Duff, Erisman & Reid [35]. Among the many papers that deal with effects of orderings on preconditioned iterative methods (in particular Krylov subspace solvers) we mention [14],[23],[29],[31],[36],[37],[89].

There are several possible “guidelines” for picking orderings. Popular ones are orderings for small bandwidth, which attempt to minimize the profile of the matrix (e.g. Cuthill-McKee [28] or Reverse Cuthill-McKee [52]), or orderings which attempt to minimize the amount of fill-in, i.e. the number of entries that were initially zero and become nonzero in the process of elimination (e.g. Minimum Degree [108]). A convenient way to obtain such orderings is by working with the graph of the matrix [52]. Ordering strategies which consider the graph of the matrix as well as its values are MDF [29] and TPABLO [83].

In seeking an effective ordering technique for our reduced matrix, our purpose in general is to choose a strategy that results in a well-structured narrow-banded matrix. In addition, we are interested in ordering the unknowns so that the main diagonal blocks of the matrix are as dense as possible. Some motivation for that can be given, for example, by considering the following result due to Varga [113, Thm 3.15]:

Let $A = M_1 - N_1 = M_2 - N_2$ be two regular splittings of $A$, where $A^{-1} > 0$. If $N_2 \geq N_1 \geq 0$,
equality excluded, then
\[ 0 < \rho(M_1^{-1}N_1) < \rho(M_2^{-1}N_2) < 1 \tag{3.1} \]

On the other hand, dense blocks mean more computational work when block iterative solvers are considered. For some of these solvers the block diagonal part of the matrix needs to be inverted in each iteration. There is a clear tradeoff between the number of nonzero diagonals that belong to the block diagonal part of the matrix and the amount of work required to invert this submatrix. Of course, the term “block diagonal part” is very loose, as it is determined by the user (taking things to extreme, the user could decide that the bandwidth of the “block diagonal part” of the matrix is equal to the matrix bandwidth). Here we shall adopt a strategy of referring only to the block diagonal part of the matrix whose bandwidth does not depend on the matrix size, and attempting to group as many nonzero diagonals as possible in this part of the matrix. Inversion of the diagonal block would then still require only a number of floating point operations which is proportional to the number of unknowns, and at the same time a substantial part of the matrix is to be included in its block diagonal part.

Examining the sparsity pattern of the lexicographically ordered matrix (Fig. 2.5) reveals the flaws of this ordering: it is not suitable for the reduced grid, as it does not take into account the structure of the grid, that is, the fact that the red points are now missing. In addition, it does not take into account the special structure of the computational molecule of the reduced operator. Since the stencil is not compact, and contains grid points from five parallel planes, an ordering strategy which numbers unknowns from more than one plane at a time might be required.

Another major point of consideration should be the ability to parallelize the solution procedures: the matrices considered here are very large due to the number of space dimensions, and their block structure might lend itself to parallelism.

Below we present ordering strategies which can be applied to any three-dimensional problem. The idea is to divide the unknowns into one-dimensional and two-dimensional sets, and
analyze the ordering not only relative to the three-dimensional grid but also relative to the lower dimensional grid of sets of gridpoints.

3.1 Block Ordering Strategies for 3D Grids

Denote the number of unknowns by $N$ and suppose that a certain gridpoint’s index $\ell$, $1 \leq \ell \leq N$, is associated with coordinate values $x$, $y$, $z$. We denote its coordinate indices by $i = \frac{x}{h}$, $j = \frac{y}{h}$, $k = \frac{z}{h}$, and refer to sets of coordinate indices simply as “coordinate sets”. The coordinate indices are integers assuming values from 1 to $n$. Here $n$ is the same as in the previous chapters.

**Definition 3.1.** Let $\{S_m\}$ be $O(n)$-item disjoint sets of clustered indices, such that the coordinate set of the gridpoints contained in each set $S_m$ contains all integers from 1 to $n$ for exactly one of the independent variables, and $\bigcup S_m = \{1, \ldots, N\}$. We call each of the sets $S_m$ a 1D block of gridpoints, and we say that this block is $x$-oriented, $y$-oriented or $z$-oriented, according to which independent variable is associated with all integers from 1 to $n$.

**Definition 3.2.** Let $\{T_m\}$ be $O(n^2)$-item disjoint sets of clustered indices, such that the coordinate set of the gridpoints contained in each set $T_m$ contains all integers from 1 to $n$ for two of the independent variables, and $\bigcup T_m = \{1, \ldots, N\}$. Each set $T_m$ is a 2D block of gridpoints, and we say that it is either $x$-$y$ oriented, $x$-$z$ oriented, or $y$-$z$ oriented, according to which two independent variables are associated with all integers from 1 to $n$ in the coordinate set.

Notice that from the above definitions it follows that the 1D blocks do not necessarily form single lines, and the 2D blocks do not necessarily form single planes. Below we define a 2D block grid and a block computational molecule.

**Definition 3.3.** Suppose the one-dimensional blocks associated with a certain ordering are $s_1$-oriented, where $s_1$ is one of $\{x, y, z\}$. Then a three-dimensional grid of points can be redefined as a 2D block grid of $s_1$-oriented 1D blocks. The variables associated with this grid are $\{s_2, s_3\} = \{x, y, z\}\setminus \{s_1\}$.

**Definition 3.4.** For a certain given gridpoint, its associated block computational molecule is defined as the computational molecule in the corresponding 2D block grid. That is, its com-
Chapter 3. Ordering Strategies

Components are all the sets of 1D blocks which contain at least one gridpoint which belongs to the point computational molecule associated with the 3D problem.

Using the above definitions, we can now easily define variants of block orderings:

**Definition 3.5.** A certain 3D ordering strategy is called *natural block ordering* if the one-dimensional blocks are ordered in the corresponding 2D block grid using natural lexicographic ordering.

**Definition 3.6.** A certain 3D ordering strategy is called *red/black block ordering* if the one-dimensional blocks are ordered in the corresponding 2D block grid using red/black ordering.

**Definition 3.7.** A certain 3D ordering strategy is called *toroidal block ordering* if the one-dimensional blocks are ordered in the corresponding block 2D grid using toroidal ordering strategy [41].

In Fig. 3.1 we illustrate the above-mentioned orderings. See [41] for an explanation on toroidal ordering and its advantages.

![Figure 3.1](image)

Figure 3.1: orderings of the 2D block grid. Each point in these grids corresponds to a 1D block of gridpoints in the underlying 3D grid.

Having established the general idea, we now consider particular families of orderings for the reduced grid. Various families can be determined according to the way their 1D and 2D sets of gridpoints are picked.


3.2 The Family of Two-Plane Orderings

This ordering corresponds to ordering the unknowns by gathering blocks of \(2n\) gridpoints from two horizontal lines and two adjacent planes. In Figure 3.2 three different members of the *natural* two-plane ordering are depicted. For notational convenience, we label this family by "2PN" ("2P" stands for two-plane, and "N" stands for natural). Similarly, we name the red/black family and the toroidal family by "2PRB" and "2PT" respectively. Two additional letters are added in order to distinguish between members of the family: for example 2PNxy is the ordering associated with 1D blocks of gridpoints in \(x\)-direction and 2D blocks in \(x-y\) direction, and so on.

![Figure 3.2: three members in the family of natural two-plane orderings](image)

Let us illustrate the definitions presented in the previous section by examining the ordering 2PNxy: the basic one-dimensional sets are \(2n\)-item sets: for the specific case of Fig. 3.2(a), which corresponds to \(n = 4\), indices 1-8, 9-16, 17-24 and 25-32 are each a one-dimensional set which forms a "stripe" of size \(n \times 2 \times 2\). Thus the 1D sets are \(x\)-oriented. Next, the sets 1-16, 17-32 form an \(n \times n \times 2\) shape, thus the 2D sets are \(x-y\) oriented. The block grid is thus a 2D grid of \(x\)-oriented 1D sets of gridpoints, associated with the independent variables \(y\) and \(z\), and if we associate \(y\) with rows and \(z\) with columns, then the 1D sets are ordered in a natural lexicographic rowwise fashion.
Chapter 3. Ordering Strategies

In Fig. 3.3 we depict the two-plane red/black and toroidal families of orderings. The combination of Fig. 3.1, Fig. 3.2 and 3.3 clarifies how the ordering for a larger grid is done.

Figure 3.3: red/black and toroidal two-plane ordering corresponding to $x$-$y$ oriented 2D blocks

We now specify the entries of the matrices for the two-plane family of orderings. We arbitrarily pick one member of this family, 2PNxz; any other family member’s specific entries can be deduced simply by interchanging the roles of $x$, $y$ and $z$ or by changing the indexing of the blocks in an obvious manner.

The matrix can be written as an $\frac{n}{2}$-order block tridiagonal matrix

$$S = \text{tri}[S_{j,j-1}, S_{j,j}, S_{j,j+1}] .$$

Each of the above components of $S$ is an $n^2 \times n^2$ matrix, block tridiagonal relative to $2n \times 2n$ blocks. Let us use superscripts $(-1)$, $(0)$ and $(1)$ to describe subdiagonal, diagonal, and superdiagonal blocks of each of the block matrices, respectively. Denote by ’cntr’ the center of the computational molecule, specified by (2.6) and the explanation that follows.

The diagonal submatrices $S_{j,j}^{(0)}$ have nonzero entries in diagonals -4 to 4, and their diagonals
Chapter 3. Ordering Strategies

are given as follows:

\[-e^2, -2ef \cdot E_{100}, -2ee \cdot E_{1001} - 2bc \cdot E_{0110}, -2eg \cdot E_{01}, -2ef \cdot E_{1000} - 2bf \cdot E_{0010},
\]
\[\text{cntr, } -2bg \cdot E_{0100} - 2df \cdot E_{10} - 2eg \cdot E_{0011} - 2bd E_{0110} - 2de \cdot E_{1001}, -2dg \cdot E_{01}, -d^2].\]

(3.3)

The other submatrices contained in \(S_{j,j}\) are of an irregular structure. The superdiagonal submatrix \(S_{j,j}^{(1)}\) has nonzero entries in diagonals -3 to 1, as follows:

\[[-2eg \cdot E_{10}, 0, -2egE_{1000} - 2bg \cdot E_{0010}, -g^2, -2dg \cdot E_{10}] ,\]

(3.4)

and the subdiagonal submatrix \(S_{j,j}^{(-1)}\) has nonzero entries in diagonals -1 to 3, as follows:

\[[-2ef \cdot E_{01}, -f^2, -2bf \cdot E_{0100} - 2ef \cdot E_{0001}, 0, -2df \cdot E_{01}] .\]

(3.5)

The components of \(S_{j,j+1}\) and \(S_{j,j-1}\) are given by:

\[S_{j,j+1}^{(-1)} = \text{diag}[-2ef \cdot E_{0100}] ;\]

(3.6)

\[S_{j,j+1}^{(0)} = \text{penta}[-2ee \cdot E_{0110}, -2ef \cdot E_{0010}, -e^2, -2eg \cdot E_{0100}, -2de E_{0110}] ;\]

(3.7)

\[S_{j,j+1}^{(1)} = \text{diag}[-2eg \cdot E_{0010}] ;\]

(3.8)

\[S_{j,j-1}^{(-1)} = \text{diag}[-2bf \cdot E_{0001}] ;\]

(3.9)

\[S_{j,j-1}^{(0)} = \text{penta}[-2bc \cdot E_{1001}, -2bf \cdot E_{1000}, -b^2, -2bg \cdot E_{0001}, -2bdE_{1001}] ;\]

(3.10)

\[S_{j,j-1}^{(1)} = \text{diag}[-2bg \cdot E_{100}] .\]

(3.11)

The sparsity pattern of the matrix is depicted in Fig. 3.4(a). Having specified the entries of the matrix, we can now find the block computational molecule corresponding to the two-plane ordering. Refer to the natural ordering depicted in Fig. 3.2. Examining the 2n-item blocks of the ordering, we get a block computational molecule of the form depicted in Fig.
Chapter 3. Ordering Strategies

Figure 3.4: sparsity patterns of two members of the two-plane family of orderings

3.5(c). Fig. 3.5(a) and 3.5(b) illustrate what sets are associated with a single gridpoint. As is evident, the structure depends upon the parity of one of the gridpoint’s indices. The block computational molecule is obtained by taking the union of the $2n$-item sets associated with each of the gridpoints in the block, and is identical to the computational molecule of the classical nine-point operator. Revealing this structure allows one to learn about the cyclically reduced operator (as a block operator), by what is known about the 9-point operator. For example, it is clear that the reduced matrix does not have block Property A relative to partitioning into $2n \times 2n$ blocks. (discussion of this is to come in Sec. 3.4.)

Figure 3.5: block computational molecule corresponding to the family of orderings 2PN

The 9-point operator gives rise to matrices for which parallelization can be obtained by using four-color schemes [57],[95]. In general, multicoloring techniques are useful for parallel computation [95]. Here we consider four-color ordering of 1D sets of gridpoints, and the resulting
Chapter 3. Ordering Strategies

Matrix for the two-plane ordering is illustrated in Fig. 3.6. Notice that the main blocks are narrow-banded block diagonal matrices, and thus are easy to invert.

![Figure 3.6: four-color 1D block version of the two-plane ordering (the matrix is of size 2048 x 2048, obtained by discretization on a 16 x 16 x 16 grid)](image)

The same ideas can be applied to 2D blocks of grid points. The block grid in this case is one-dimensional. In Fig. 3.4(b) the sparsity pattern of a 256 x 256 matrix corresponding to red/black ordering based on 2D blocks is depicted. As in the common cases of red/black ordering for problems with a smaller number of space dimensions, the bandwidth of the red/black matrix is larger than the one corresponding to natural ordering.

3.3 The Family of Two-Line Orderings

The two-plane ordering, presented in the previous section, is unique to three-dimensional problems. We now consider an alternative: a straightforward generalization to 3D of the two-line ordering for 2D problems used by Elman & Golub in [41]. In this ordering the numbering is done in pairs of horizontal lines that lie in a single plane.

The ordering and sparsity structure of the matrix corresponding to the natural two-line ordering 2L.Nxy are depicted in Fig. 3.7. Other members of the family can be obtained in a straightforward manner. Here the 1D blocks are sets of \( n \) grid points, and each of the 2D blocks corresponds to a single \( x-y \) plane. The reduced matrix for this ordering strategy can be written
Chapter 3. Ordering Strategies

![Diagram](image)

(a) ordering  (b) sparsity pattern

Figure 3.7: ordering and the sparsity pattern of the matrix associated with the 2LNxy ordering strategy

as a block pentadiagonal matrix:

\[ S = \text{penta}[S_{j,j-2}, S_{j,j-1}, S_{j,j}, S_{j,j+1}, S_{j,j+2}]. \]  

(3.12)

Each \( S_{i,j} \) is \((n^2/2) \times (n^2/2)\), and is a combination of \(n^2/2\) uncoupled matrices, each of size \(n \times n\). The diagonal matrices \( \{S_{j,j}\} \) are themselves block tridiagonal. Each submatrix is of size \(n \times n\) and its diagonal block is

\[
S_{j,j}^{(0)} = \begin{cases} 
\text{penta}(-e^2, -2bc \cdot E_{01} - 2ce \cdot E_{10}, \text{cnt}, -2bd \cdot E_{01} - 2de \cdot E_{10}, -d^2) & j \text{ odd} \\
\text{penta}(-e^2, -2bc \cdot E_{10} - 2ce \cdot E_{01}, \text{cnt}, -2bd \cdot E_{10} - 2de \cdot E_{01}, -d^2) & j \text{ even} 
\end{cases}
\]

For the superdiagonal and the subdiagonal blocks of the matrices \( S_{j,j} \) we have the following irregular tridiagonal structure, which depends on whether \( j \) is even or odd. The superdiagonal matrices are given by

\[
S_{j,j}^{(1)} = \begin{pmatrix}
-e^2 & -2de \\
-e^2 & -2ce & -e^2 & -2de \\
-2ce & -e^2 & -2de & \ddots \\
\end{pmatrix}
\]

(3.13)
Chapter 3. Ordering Strategies

if $j$ is odd, or

$$S_{j,j}^{(1)} = \begin{pmatrix} -e^2 \\ -2ce & -e^2 & -2de \\ & \ddots & \ddots \\ & & -e^2 \\ & & & -2ce & -e^2 \end{pmatrix}$$

(3.14)

if $j$ is even.

The subdiagonal matrices are

$$S_{j,j}^{(-1)} = \begin{pmatrix} -b^2 \\ -2bc & -b^2 & -2bd \\ & \ddots & \ddots \\ & & -b^2 \\ & & & -2bc & -b^2 \end{pmatrix}$$

(3.15)

if $j$ is odd, or

$$S_{j,j}^{(-1)} = \begin{pmatrix} -b^2 & -2bd \\ -b^2 \\ -2bc & -b^2 & -2bd \\ & \ddots & \ddots \\ & & & -2bc & -b^2 & -2bd \\ & & & & \ddots \\ & & & & & -b^2 \end{pmatrix}$$

(3.16)

if $j$ is even.
Chapter 3. Ordering Strategies

The superdiagonal and the subdiagonal blocks of $S_j$, $S_{j,j\pm1}$, are block tridiagonal:

$$S^{(-1)}_{j,j-1} = \begin{cases} 
\text{diag}(-2bf \cdot E_{01}) & j \text{ odd} \\
\text{diag}(-2bf \cdot E_{10}) & j \text{ even}
\end{cases} \quad (3.17)$$

$$S^{(0)}_{j,j-1} = \begin{cases} 
\text{tri}(-2cf, -2bf \cdot E_{10} - 2cf \cdot E_{01}, -2df) & j \text{ odd} \\
\text{tri}(-2cf, -2bf \cdot E_{01} - 2cf \cdot E_{10}, -2df) & j \text{ even}
\end{cases} \quad (3.18)$$

$$S^{(1)}_{j,j-1} = \begin{cases} 
\text{diag}(-2cf \cdot E_{10}) & j \text{ odd} \\
\text{diag}(-2cf \cdot E_{01}) & j \text{ even}
\end{cases} \quad (3.19)$$

$$S^{(-1)}_{j,j+1} = \begin{cases} 
\text{diag}(-2bg \cdot E_{01}) & j \text{ odd} \\
\text{diag}(-2bg \cdot E_{10}) & j \text{ even}
\end{cases} \quad (3.20)$$

$$S^{(0)}_{j,j+1} = \begin{cases} 
\text{tri}(-2cg, -2bg \cdot E_{10} - 2cg \cdot E_{01}, -2dg) & j \text{ odd} \\
\text{tri}(-2cg, -2bg \cdot E_{01} - 2cg \cdot E_{10}, -2dg) & j \text{ even}
\end{cases} \quad (3.21)$$

$$S^{(1)}_{j,j+1} = \begin{cases} 
\text{diag}(-2cg \cdot E_{10}) & j \text{ odd} \\
\text{diag}(-2cg \cdot E_{01}) & j \text{ even}
\end{cases} \quad (3.22)$$

Finally, the matrices $S_{j,j-2}$ and $S_{j,j+2}$ are diagonal:

$$S_{j,j-2} = \text{diag}(-f^2), \quad j = 3, \ldots, n; \quad (3.23)$$

$$S_{j,j+2} = \text{diag}(-g^2), \quad j = 1, \ldots, n-2. \quad (3.24)$$

The block computational molecule corresponding to the two-line ordering relative to splitting into 1D blocks is presented in Fig. 3.8.
Chapter 3. Ordering Strategies

![Diagram](image)

(a) even \( k = \frac{n}{2} \)  
(b) odd \( k = \frac{n}{2} \)  
(c) computational molecule

Figure 3.8: block computational molecule corresponding to the ordering strategy 2L_Nxy

3.4 Comparison Results

In order to compare the various orderings, we consider stationary methods, for which there are some useful general results [113] which make it possible to relate the orderings to the rate of convergence. Various splittings are possible. We consider two obvious ones, based on dimension: the term "one-dimensional splitting" is used for a splitting which is based on partitioning the matrix into \( O(n) \) blocks (\( 2n \times 2n \) blocks for the two-plane ordering and \( n \times n \) blocks for the two-line ordering). A "two-dimensional splitting" is one which is based on partitioning the matrix into \( O(n^2) \) blocks (\( n^2 \times n^2 \) blocks for the two-plane ordering and \( (n^2/2) \times (n^2/2) \) blocks for the two-line ordering). In other words, the 1D (2D) splitting for both ordering strategies is essentially associated with 1D (2D) sets of gridpoints. The partitionings for the two-plane matrix are illustrated in Fig. 3.9.

An illustration of the importance of using ordering strategies which fit the structure of the reduced grid is given in Fig. 3.10: the sparsity patterns of a single \( n^2 \times n^2 \) diagonal block of the two-plane matrix and the matrix that corresponds to point natural lexicographic ordering are depicted. The matrices in the figure arise from discretization on a \( 12 \times 12 \times 12 \) grid. If the two matrices are referred to as block tridiagonal, the bandwidth of the block diagonal part of the two-plane matrix is significantly smaller. At the same time, if only diagonals whose indices do not depend on \( n \) are referred to, then the two-plane matrix has more of them grouped next to the main diagonal.
In Fig. 3.11 the spectral radii of the one-dimensional Jacobi iteration matrices for both of the two-plane ordering and the natural lexicographic ordering are plotted. The graphs correspond to a constant coefficient case, and two cross-sections of the mesh Reynolds numbers are given. As is evident, the two-plane iteration matrix has a smaller spectral radius. This results in faster convergence.

It is clear, then, that block ordering strategies such as the two-plane ordering can be more effective than natural lexicographic ordering of the reduced grid. At this point, we compare the two families of block orderings that have been introduced in the previous sections: the two-plane ordering and the two-line ordering.

**Theorem 3.1.** If the reduced matrix is an $M$-matrix, then for both the 1D and the 2D partitionings the spectral radius of the Jacobi iteration matrix associated with two-plane ordering is smaller than the spectral radius of the iteration matrix associated with the two-line ordering.

**Proof.** Each of the two ordering strategies generates a matrix which is merely a symmetric permutation of a matrix associated with the other ordering. Suppose $S_1 = M_1 - N_1$ is either a 1D splitting or a 2D splitting of the two-plane ordering matrix, and $S_2 = M_2 - N_2$ is an analogous splitting for the two-line ordering, with 1D and 2D blocks of gridpoints oriented in the same direction. There exists a permutation matrix $P$ such that $P^T S_2 P = S_1$. Consider the
Chapter 3. Ordering Strategies

Figure 3.10: a zoom on 2D blocks of the matrices corresponding to two ordering strategies of the reduced grid

splitting $P^T S_2 P = P^T M_2 P - P^T N_2 P$. It is straightforward to show by examining the matrix entries that $P^T N_2 P \geq N_1$. The latter are both nonnegative matrices, thus by [113, Thm 3.15] it follows that $0 < \rho(M_1^{-1} N_1) < \rho(P^T M_1^{-1} N_2 P) = \rho(M_1^{-1} N_2) < 1$.

Some of our numerical experiments, presented in Fig. 3.12, validate the results indicated in Thm. 3.1. In the figure spectral radii of the block Jacobi iteration matrices are presented. It is interesting to observe that the spectral radius of the two-plane iteration matrix is smaller also for the case $bc, cd, fg < 0$, which corresponds to the region of mesh Reynolds numbers larger than 1 and for which the theorem does not apply. A few cross-sections of mesh Reynolds numbers are examined. For example, graph (a) corresponds to flow with the same velocity in $x, y$ and $z$ directions. Graph (b) corresponds to flow only in $x$ and $y$ directions, and no convection in the $z$ direction, and so on [see Eq. (1.16) for the definition of $\beta, \gamma$ and $\delta$].

We note that the bandwidth of the two-plane matrix is larger than the bandwidths of the natural lexicographically ordered matrix and the two-line matrix: $n^2 + 2n$ vs. $n^2$. However, the difference is negligible, as typically $2n \ll n^2$. Smaller bandwidth can be obtained by using direct solver orderings such as RCM (see Fig. 3.13). However, numerical experiments that have been performed with this ordering show slower convergence of block stationary methods,
Figure 3.11: comparison of the spectral radii of block Jacobi iteration matrices for certain cross-sections of the mesh Reynolds numbers. The upper curves correspond to lexicographic ordering. The lower curves correspond to two-plane ordering. The matrices were obtained using centered difference discretization.

compared to the two-plane ordering,

Next, we consider the issue of consistent ordering. Recall the definition of consistently ordered matrices (see [67],[95],[117]):

**Definition 3.8.** A matrix is said to be consistently ordered if the vertices of its adjacency graph can be partitioned in $p$ sets $S_1$, $S_2$, ..., $S_p$ with the property that any two adjacent vertices $i$ and $j$ in the graph belong to two consecutive partitions $S_k$ and $S_{k'}$, with $k' = k - 1$ if $j < i$, and $k' = k + 1$ if $j > i$.

A matrix that is consistently ordered has Property $A$; conversely, a matrix with Property $A$ can be permuted so that it is consistently ordered [95]. It should be noted that not all consistently ordered matrices for a given problem have the same Jordan canonical form [47]. This is of importance, because the size of the Jordan block associated with the largest eigenvalue (in magnitude) affects the magnitude of the spectral norm of the iteration matrix; the latter affects the speed of convergence [47],[113].

It was mentioned in the Introduction that the unreduced matrix has Property $A$ relative
Figure 3.12: Spectral radii of iteration matrices vs. mesh Reynolds numbers for the block Jacobi scheme, using 1D splitting and centered difference discretization. The broken lines correspond to two-plane ordering. The solid lines correspond to two-line ordering.

to partitioning into $1 \times 1$ matrices. For the reduced system, we have the following observations:

**Proposition 3.1.** The reduced matrix associated with two-line ordering, $S_L$, does not have Property A relative to 1D or 2D partitionings.

**Proof.** This can be deduced directly from the structure of the block computational molecules of the 1D and 2D partitionings. Formally, let $S_{i,j}$ denote the $(i,j)$th $n \times n$ block of $S_L$, and let $Q$ be an $(n^2/2) \times (n^2/2)$ matrix, whose entries satisfy $q_{i,j} = 1$ if $S_{i,j} \neq 0$ and $q_{i,j} = 0$ otherwise. Let $T$ be an $n \times n$ matrix, such that $t_{i,j} = 1$ if the $(i,j)$th $(n^2/2) \times (n^2/2)$ block submatrix of $S$ is nonzero, and $t_{i,j} = 0$ otherwise. $T$ is a pentadiagonal matrix, thus does not have Property A. Since $Q$ could have Property A only if $T$ did, it does not have Property A either. \hfill $\square$

**Proposition 3.2.** The reduced matrix associated with two-plane ordering, $S_P$, does not have Property A relative to 1D partitioning.

**Proof.** Let $S_{i,j}$ denote the $(i,j)$th $(2n \times 2n)$ block of $S_P$, and let $Q$ to be an $(n^2/4) \times (n^2/4)$ matrix,
Chapter 3. Ordering Strategies

Figure 3.13: symmetric reverse Cuthill-McKee ordering of the reduced matrix (of size $256 \times 256$).

whose entries satisfy $q_{i,j} = 1$ if $S_{i,j} \neq 0$ and $q_{i,j} = 0$ otherwise. It is straightforward to see
that the nonzero pattern of $Q$ is identical to that of the matrix associated with using a 9-point
operator for a 2D grid (indeed, this is indicated by the structure of the block computational
molecule). Since the latter does not have Property A relative to partitioning into $1 \times 1$ matrices,
the result follows.

On the other hand, we have:

**Proposition 3.3.** The reduced matrix associated with two-plane ordering, $S_P$, has Property A,
and moreover - is consistently ordered relative to 2D partitioning.

**Proof.** The matrix is block tridiagonal relative to this partitioning, thus is consistently ordered
[117].

The above results indicate that it will be difficult to analyze the convergence properties of
the reduced matrices, as they are not necessarily consistently ordered. In Chapter 4 we will
address this point and provide some analysis which overcomes the loss of Property A for the
two-plane matrix, relative to 1D partitioning.
Chapter 4
Convergence Analysis for Block Stationary Methods

In this chapter bounds on the convergence rates of the Jacobi, Gauss-Seidel, and SOR schemes are derived. First, the constant coefficient case is analyzed, and then the results are generalized to the variable coefficient case. We consider block splittings, which well suit the block structure of the reduced matrix. In particular, the 1D and the 2D splittings discussed in Chapter 3 are considered. Since the two-plane ordering was shown in Chapter 3 to be more effective than other orderings, we focus on matrices associated with this ordering. We analyze the block Jacobi scheme, derive upper bounds on convergence rates, and demonstrate their tightness. We then discuss consistently ordered matrices [113],[117] in the context of the reduced matrix, and use our observations to derive bounds for the block Gauss-Seidel and SOR schemes. Finally, convergence results are derived for the unreduced (original) system, and are compared to the bounds on convergence rates for the reduced system.

4.1 Symmetrization of the Reduced System

In order to estimate the spectral radii of the iteration matrices involved in solving the reduced system, the following idea of Elman & Golub [40],[41] is used: denoting the reduced matrix by $S$, suppose there exists a real diagonal nonsingular matrix $Q$, such that $\hat{S} = Q^{-1}SQ$ is symmetric. The symmetry can then be used as follows: suppose $S = D - C$ is a splitting, such that $\hat{D} = Q^{-1}DQ$ is symmetric positive definite, and $\hat{C} = Q^{-1}CQ$ is symmetric. Since $D^{-1}C$
and $\hat{D}^{-1}\hat{C}$ are similar, it follows that:

$$\rho(D^{-1}C) = \rho(\hat{D}^{-1}\hat{C}) \leq \|\hat{D}^{-1}\|_2\|\hat{C}\|_2 = \frac{\rho(\hat{C})}{\lambda_{\min}(\hat{D})}. \quad (4.1)$$

The latter is useful, since the eigenvalues of a symmetric matrix are real, are positive if the matrix is positive definite, and it is easier to compute them (compared to nonsymmetric matrices).

4.1.1 The Constant Coefficient Case

Consider the model problem (1.15). Let $a, b, c, d, e, f, g$ be the components of the computational molecule [see Eq. (1.17), (1.18) and (1.19)]. We have:

**Theorem 4.1.** The reduced matrix $S$ can be symmetrized with a real diagonal similarity transformation if and only if the products $bde, bef$ and $cdfg$ are positive.

**Proof.** For a certain ordering, the corresponding matrix is merely a symmetric permutation of a matrix which is associated with another ordering. Thus we can examine the symmetrization conditions for the matrix associated with a particular ordering without loss of generality. We thus refer to the particular ordering whose associated matrix was described in detail in Chapter 3, namely 2PNxz. Our aim is to find a real diagonal matrix $Q$, so that $Q^{-1}SQ$ is symmetric. Suppose $Q = \text{diag}[Q_{1,1}, Q_{1,2}, \ldots, Q_{1,6}, Q_{2,1}, Q_{2,2}, \ldots, Q_{2,6}, Q_{2,1}, Q_{2,2}, \ldots, Q_{2,6}, Q_{2,1}, Q_{2,2}, \ldots, Q_{2,6}]$, where each matrix $Q_{j,l}, 1 \leq j, l \leq \frac{n}{2}$, is a diagonal $2n \times 2n$ matrix whose entries are denoted by:

$$Q_{j,l} = \text{diag}[q_{j,1}^{(j,l)}, \ldots, q_{2n}^{(j,l)}]. \quad (4.2)$$

Consider first the diagonal block matrices $S_{j,j}^{(0)}$. We require that $Q_{j,l}^{-1}S_{j,j}^{(0)}Q_{j,l}$ be symmetric. Notice that for any $j$, $S_{j,j}^{(0)}$ is a notation that corresponds to $\frac{n}{2}$ submatrices, but in order to avoid additional notation, we do not add a script that indicates the location of these matrices in each block, as it can be understood from the double index used for the submatrices of $Q$. A straightforward computation for the entries of $S_{j,j}^{(0)}, 1 \leq j \leq \frac{n}{2}$ and for all the $\frac{n}{2}$ submatrices in
Chapter 4. Convergence Analysis for Block Stationary Methods

each block $S_{j,j}$, leads to the following conditions:

\[
\left( \frac{q_i^{(j,l)}}{q_i^{(j,l-1)}} \right)^2 = \frac{c^2}{d^2}, \quad i = 5, \ldots, 2n, \quad (4.3)
\]

\[
\left( \frac{q_i^{(j,l)}}{q_i^{(j,l-3)}} \right)^2 = \frac{cf}{dg}, \quad i = 4, 6, 8, \ldots, 2n, \quad (4.4)
\]

\[
\left( \frac{q_i^{(j,l)}}{q_i^{(j,l-2)}} \right)^2 = \frac{ce}{bd}, \quad 1 \leq i \leq 2n, \quad i \text{ mod } 4 = 2 \text{ or } 3, \quad (4.5)
\]

\[
\left( \frac{q_i^{(j,l)}}{q_i^{(j,l-2)}} \right)^2 = \frac{be}{df}, \quad 1 \leq i \leq 2n, \quad i \text{ mod } 4 = 0 \text{ or } 1, \quad (4.6)
\]

\[
\left( \frac{q_i^{(j,l)}}{q_i^{(j,l-1)}} \right)^2 = \frac{ce}{bd}, \quad i = 3, 5, 7, \ldots, 2n - 1, \quad (4.7)
\]

\[
\left( \frac{q_i^{(j,l)}}{q_i^{(j,l-1)}} \right)^2 = \frac{bf}{eg}, \quad 1 \leq i \leq 2n, \quad i \text{ mod } 4 = 2, \quad (4.8)
\]

\[
\left( \frac{q_i^{(j,l)}}{q_i^{(j,l-1)}} \right)^2 = \frac{bf}{eg}, \quad 1 \leq i \leq 2n, \quad i \text{ mod } 4 = 0. \quad (4.9)
\]

The restriction that the diagonal similarity transformation be real leads to the following conditions:

- Equations (4.4) and (4.7) imply $cdef > 0$.
- Equations (4.5) and (4.6) imply $bcde > 0$.
- Equations (4.8) and (4.9) imply $bfg > 0$.

We choose $q_i^{(j,l)}, 1 \leq j, l \leq \frac{n}{4}$ arbitrarily, and use Equations (4.7), (4.8) and (4.9) to determine $q_i^{(j,l)}, 2 \leq i \leq 2n, 1 \leq j, l \leq \frac{n}{4}$. In so doing, we must make sure that Equations (4.3)-(4.6) are consistent with Equations (4.7)-(4.9). Indeed there is full consistency. Below we show
this for entries whose index, $i$, satisfies $i \mod 4 = 0$. The same procedure can be done for the other values of $i$. Since $(i - 1) \mod 4 = 3$, applying (4.7) for $i - 1$ (rather than $i$) and multiplying it by (4.9) we obtain \( \left( \frac{q_{i-1}^{(j,i)}}{q_{i-1}^{(j,i)}} \right)^2 = \frac{b f}{e g} \cdot \frac{e g}{b f} = \frac{b f}{e g} \), which is exactly (4.6). Next, since $(i - 2) \mod 4 = 2$, we can use (4.8) to conclude that \( \left( \frac{q_{i}^{(j,i)}}{q_{i}^{(j,i)}} \right)^2 = \frac{e f}{g f} \), which is identical to Equation (4.4). Finally, $(i - 3) \mod 4 = 1$, therefore (4.7) implies \( \left( \frac{q_{i+1}^{(j,i)}}{q_{i}^{(j,i)}} \right)^2 = \frac{f^2}{g^2} \). Multiplying this by Equation (4.4) yields \( \left( \frac{q_{i}^{(j,i+1)}}{q_{i}^{(j,i)}} \right)^2 = \frac{f^2}{g^2} \cdot \frac{e f}{g f} = \frac{e^2}{f^2} \), which is identical to (4.3). This completes the proof of consistency for indices which correspond to Equation (4.9). For indices which satisfy either Eq. (4.7) or Eq. (4.8) the process is completely analogous, and the algebraic details are omitted.

The same procedure repeats when considering the off-diagonal matrices of the main block diagonals, namely $S_{j,j}^{(\pm 1)}$, and the off-diagonal block matrices $S_{j,j+1}$. For the former we have the following equation, which determines the ratio between $q_i^{(j,i+1)}$ and $q_i^{(j,i)}$, thus defines the values of $q_i^{(j,i+1)}$, using $q_i^{(j,i)}$:

\[
\left( \frac{q_i^{(j,i+1)}}{q_i^{(j,i)}} \right)^2 = \frac{f^2}{g^2} , \quad 1 \leq i \leq 2n .
\]

(4.10)

Notice that (4.10) establishes conditions for values that were previously considered arbitrary, namely $q_1^{(j,i)}$, $l > 1$. At this stage only $q_1^{(j,1)}$, $1 \leq j \leq \frac{n}{2}$ are left arbitrary.

In addition, we have the following: (In the equations that appear below, $l$ goes from 1 to $\frac{n}{2} - 1$, and $j$ goes from 1 to $\frac{n}{2}$.)

\[
\left( \frac{q_{i-1}^{(j,i+1)}}{q_i^{(j,i)}} \right)^2 = \frac{b f}{e g} , \quad 1 \leq i \leq 2n , \quad i \mod 4 = 2 ,
\]

(4.11)

\[
\left( \frac{q_{i}^{(j,i+1)}}{q_{i-1}^{(j,i)}} \right)^2 = \frac{e f}{d g} , \quad i = 3, 5, \ldots , 2n - 1 .
\]

(4.12)
Equations (4.11)-(4.14) can all be obtained from combinations of the group of Equations (4.3)-(4.10), thus are consistent and do not impose any additional restrictions.

Next, the off-diagonal block matrices $S_{j,j \neq 1}$ are examined. We start with $S^{(0)}_{j,j \neq 1}$. The following conditions, for $2 \leq j \leq \frac{n}{2}$ and $1 \leq l \leq \frac{n}{2}$, need to be satisfied:

$$
\left( \frac{q_{i}^{(j+1,l)}}{q_{i}^{(j,l)}} \right)^2 = \frac{b^2}{e^2}, \quad 1 \leq i \leq 2n, \quad i \mod 4 = 0, \quad (4.15)
$$

$$
\left( \frac{q_{i}^{(j+1,l)}}{q_{i-1}^{(j,l)}} \right)^2 = \frac{b f}{e g}, \quad 1 \leq i \leq 2n, \quad i \mod 4 = 2, \quad (4.16)
$$

$$
\left( \frac{q_{i}^{(j+1,l)}}{q_{i-2}^{(j,l)}} \right)^2 = \frac{b c}{c d}, \quad 1 \leq i \leq 2n, \quad i \mod 4 = 2 \text{ or } 3, \quad (4.17)
$$

$$
\left( \frac{q_{i}^{(j+1,l)}}{q_{i}^{(j,l)}} \right)^2 = \frac{b g}{e f}, \quad 1 \leq i \leq 2n, \quad i \mod 4 = 0, \quad (4.18)
$$

$$
\left( \frac{q_{i-2}^{(j+1,l)}}{q_{i}^{(j,l)}} \right)^2 = \frac{b d}{e c}, \quad 1 \leq i \leq 2n, \quad i \mod 4 = 0 \text{ or } 1, \quad (4.19)
$$

Equation (4.15) defines the values of $q_{i}^{(j+1,l)}$, $1 \leq j \leq \frac{n}{2} - 1$, given $q_{i}^{(j,l)}$. Imposing this condition leaves only $q_{1}^{(1,1)}$ arbitrary. Eq. (4.16)-(4.19) are consistent with the previous equations:
Chapter 4. Convergence Analysis for Block Stationary Methods

- Equations (4.15)+(4.8) imply (4.16).
- Equations (4.15)+(4.5) imply (4.17).
- Equations (4.15)+(4.9) imply (4.18).
- Equations (4.15)+(4.6) imply (4.19).

In order for Equations (4.15)-(4.19) to hold with a real $Q$, we require $bcde, befg > 0$. These conditions are contained in the three conditions that were imposed when Equations (4.3)-(4.9) were discussed.

Finally, the matrices $S^{(+1)}_{j,l\pm1}$ are considered. For these we require that for $1 \leq j \leq \frac{n}{2}$:

\[
\left( \frac{q_{i,j+1,l+1}^{(j,i)}}{q_i^{(j,i)}} \right)^2 = \frac{bf}{ge}, \quad 1 \leq i \leq 2n, \quad i \mod 4 = 0, \quad 2 \leq l \leq \frac{n}{2}; \quad (4.20)
\]

\[
\left( \frac{q_i^{(j+1,l+1)}}{q_{i-1}^{(j+1,l)}} \right)^2 = \frac{bg}{ef}, \quad 1 \leq i \leq 2n, \quad i \mod 4 = 2, \quad 1 \leq l \leq \frac{n}{2} - 1. \quad (4.21)
\]

Equations (4.18)+(4.10) imply (4.20), whereas Equations (4.15)+(4.11) imply (4.21):

\[
\left( \frac{q_{i-1,j+1,l+1}}{q_i^{(j+1,l)}} \right)^2 = \left( \frac{q_{i-1,j+1,l+1}}{q_i^{(j+1,l)}} \right)^2 \cdot \left( \frac{q_{i-1,j,i}}{q_i^{(j,i)}} \right)^2 = \frac{f^2}{g^2} \cdot \frac{bg}{ef} = \frac{bf}{ge}; \quad (4.22)
\]

\[
\left( \frac{q_i^{(j+1,l-1)}}{q_{i-1}^{(j,l)}} \right)^2 = \left( \frac{q_i^{(j+1,l-1)}}{q_{i-1}^{(j+1,l)}} \right)^2 \cdot \left( \frac{q_{i-1,j,i}}{q_i^{(j,i)}} \right)^2 = \frac{bg}{ef} \cdot \frac{e^2}{e^2} = \frac{bg}{ef}. \quad (4.23)
\]

Equations (4.20) and (4.21) are to hold for real $Q$ only if $befg > 0$, which is a condition which has already been imposed.

In terms of the mesh Reynolds numbers, Theorem 4.1 leads to:

\[
69
\]
Chapter 4. Convergence Analysis for Block Stationary Methods

**Corollary 4.1.** The reduced matrix $S$ can be symmetrized with a real diagonal similarity transformation for any $\beta, \gamma, \delta > 0$ if one uses the upwind (backward) schemes, and for either $|\beta|, |\gamma|, |\delta| < 1$ or $|\beta|, |\gamma|, |\delta| > 1$ if one uses centered difference schemes.

*Proof.* For upwind schemes $cdfg = (1 + 2\beta)(1 + 2\delta)$ which is always positive for positive values of $\beta, \gamma$ and $\delta$. The same is true for $befg = (1 + 2\gamma)(1 + 2\delta)$ and $bcde = (1 + 2\beta)(1 + 2\gamma)$. For centered difference schemes $cdfg = (1 - \beta^2)(1 - \delta^2)$, $befg = (1 - \gamma^2)(1 - \delta^2)$, and $bcde = (1 - \beta^2)(1 - \gamma^2)$. For $cdfg$ to be positive, we require that either $|\beta| < 1$ and $|\delta| < 1$ or $|\beta| > 1$ and $|\delta| > 1$. If $|\beta| < 1$ and $|\delta| < 1$, then $befg > 0$ implies $|\gamma| < 1$ and then $bcde > 0$ holds as well. If $|\beta|, |\delta| > 1$, then the same argument yields $|\gamma| > 1$. \qed

In order to construct the matrix $Q$ Equations (4.7)-(4.15) are used. However, from the proof it is clear how the symmetrized matrix $Q^{-1}SQ$ looks and there is no need to construct $Q$ and actually perform the similarity transformation. Moreover, the symmetrizer might contain very large values, thus using it might cause numerical difficulties. In [40] it is shown that in the one-dimensional case, for the equation $-u'' + \sigma u' = f$ with Dirichlet boundary conditions, if the first entry of the symmetrizer is 1, and the mesh Reynolds number is between 0 and 1, the last entry of the symmetrizer goes to $e^\sigma$ as $n$ goes to infinity, and thus is very large for large values of the underlying PDE coefficient. Instability occurs in the three-dimensional problem, as is now shown:

**Theorem 4.2.** Solving the symmetrized linear system obtained by using the diagonal symmetrizer $Q$ is numerically unstable.

*Proof.* Symmetrizing by $Q$ means that instead of solving the system $Sx = w$, a linear system of the form

$$\hat{S}\hat{x} = \hat{w}, \quad (4.24)$$

is solved, where $\hat{S} = Q^{-1}SQ$ is symmetric, $\hat{x} = Q^{-1}x$ and $\hat{w} = Q^{-1}w$. The matrix $Q$ is constructed as follows:
Choose any value for $q_i^{(1,1)}$.

for $i = 2$ to $2n$

if $i \mod 2 = 1$

$$q_i^{(1,1)} = \sqrt{\frac{cg}{df}} \cdot q_{i-1}^{(1,1)}$$

if $i \mod 4 = 0$

$$q_i^{(1,1)} = \sqrt{\frac{bf}{eg}} \cdot q_{i-1}^{(1,1)}$$

if $i \mod 4 = 2$

$$q_i^{(1,1)} = \sqrt{\frac{ef}{bg}} \cdot q_{i-1}^{(1,1)}$$

for $l = 2$ to $\frac{n}{2}$

for $i = 1$ to $2n$

$$q_i^{(1,1)} = \frac{f}{g} \cdot q_i^{(1,1-1)}$$

for $j = 2$ to $\frac{n}{2}$

for $l = 1$ to $\frac{n}{2}$

for $i = 1$ to $2n$

$$q_i^{(j,1)} = \frac{b}{e} \cdot q_i^{(j-1,1)}$$

From the algorithm it follows that the entries of $Q$ are not bounded as $n$ tends to infinity. For example, the last row in the algorithm implies that if $|\frac{j}{n}| \gg 1$, even if $q_i^{(j,1)}$ is small, there exists $i$ such that computing $q_i^{(j,1)}$ will cause overflow. Such a situation occurs when $\gamma \approx 1$ and centered differences are used. In this situation, if the other two mesh Reynolds numbers are small, the original matrix is well conditioned; hence, the instability arises from the symmetrization, not as a result of conditioning of the matrix.

From Theorem 4.2 we can deduce, then, that the symmetrized matrix should be used merely for the convergence analysis; in actual numerical computation, the nonsymmetric system is used.
The entries of the symmetrizer can be determined up to sign. For example, for \(be, cd, fg > 0\), a symmetrization operator that preserves the sign of the matrix entries transforms the matrix entries as follows:

\[-b^2 \to -be; -c^2 \to -cd; -d^2 \to -ed; -e^2 \to -be; -f^2 \to -fg; -g^2 \to -fg;\]
\[-2bf \to -2\sqrt{bcfg}; -2cf \to -2\sqrt{cdfg}; -2df \to -2\sqrt{cdfg}; -2ef \to -2\sqrt{bcfg};\]
\[-2bg \to -2\sqrt{bcfg}; -2eg \to -2\sqrt{cdfg}; -2dg \to -2\sqrt{cdfg}; -2eg \to -2\sqrt{bcfg};\]
\[-2be \to -2\sqrt{bcde}; -2bd \to -2\sqrt{bcde}; -2ce \to -2\sqrt{bcde}; -2de \to -2\sqrt{bcde}.\]

The value of the computational molecule corresponding to the center point is unchanged under the symmetrization operation.

**4.1.2 The Variable Coefficient Case**

Consider Equation (2.11). As in the constant coefficient case, we denote the diagonal symmetrizer by \(Q\), and require symmetry element by element to derive the conditions of the entries of \(Q\). Let \(q_i\) denote the \(l\)th diagonal entry in \(Q\). Then

\[
\hat{s}_{j,l} = \frac{s_{j,l}^i q_l}{q_j}, \quad 1 \leq j, l \leq \frac{n^3}{2}.
\]

(4.25)

It is sufficient to look at \(2n \times 2n\) blocks. We start with the main diagonal blocks. We have to examine all the entries of the main block that appear in the \(l\)th row of the matrix, namely \(s_{l,l-4}, s_{l,l-3}, \ldots, s_{l,l}, \ldots, s_{l,l+4}\). For \(s_{l,l-4}\), if \(l \mod 2n = 5\) or is equal to 0, then \(l - 4\) corresponds to the \((i + 2, j, k)\) mesh point. Thus

\[
s_{l-4,l} = -\left. \frac{d_{\hat{a}+1,\hat{b},\hat{c}}^{\hat{a},\hat{b},\hat{c}}}{d_{\hat{a}+1,\hat{b},\hat{c}}^{\hat{a},\hat{b},\hat{c}}} \right|_{\hat{a} = i-2, \hat{b} = j, \hat{c} = k}
\]

(4.26)

and from this it follows that

\[
\left( \frac{q_l}{q_{l-4}} \right)^2 = -\left( \frac{s_{l-2,i,j,k}^i s_{l-1,i,j,k}^j s_{l-1,i,j,k}^k}{s_{l-2,i,j,k}^i s_{l-1,i,j,k}^j s_{l-1,i,j,k}^k} \right) = \frac{c_{i-1,j,k}^i c_{i,j,k}^j c_{i,j,k}^k}{d_{l-2,j,k}^i d_{l-1,j,k}^i d_{l,j,k}^i}.
\]

(4.27)

In this case the values associated with the center of the computational molecule (namely \(a_{i,j,k}\)) are canceled, but this happens only for rows that involve the \((i \pm 2, j, k)\), \((i, j \pm 2, k)\) and \((i, j, k \pm 2)\) gridpoints. Applying the same procedure for the rest of the entries of the main diagonal block we obtain
Chapter 4. Convergence Analysis for Block Stationary Methods

\[
\left( \frac{q_l}{q_{l-3}} \right)^2 = \frac{c_{l,i,j,k-1} f_{l,i,j,k}}{a_{l,i,j,k}} + \frac{c_{l,i,j,k} f_{l-1,i,j,k}}{a_{l-1,i,j,k}} \quad l \text{ mod } 2n = 0, 4, 6, \ldots, 2n - 2; \quad (4.28)
\]

\[
\left( \frac{q_l}{q_{l-2}} \right)^2 = \frac{c_{l,i,j,k+1} c_{l,i,j,k}}{a_{l,i,j,k+1}} + \frac{c_{l,i,j,k} g_{l,i,j,k}}{a_{l,i,j,k}} \quad l \text{ mod } 4 = 2 \text{ or } 3; \quad (4.29)
\]

\[
\left( \frac{q_l}{q_{l-2}} \right)^2 = \frac{b_{l,i,j,k} c_{l,i,j,k}}{a_{l-1,i,j,k}} + \frac{c_{l,i,j,k} d_{l-1,i,j,k}}{a_{l-1,i,j,k}} \quad l \text{ mod } 4 = 0 \text{ or } 1; \quad (4.30)
\]

\[
\left( \frac{q_l}{q_{l-1}} \right)^2 = \frac{d_{l-1,i,j,k} c_{l,i,j,k}}{a_{l-1,i,j,k}} + \frac{c_{l,i,j,k+1} b_{l,i,j,k+1}}{a_{l,i,j,k+1}} \quad l \text{ mod } 2n = 3, \ldots, 2n - 1; \quad (4.31)
\]

\[
\left( \frac{q_l}{q_{l-1}} \right)^2 = \frac{c_{l,i,j,k-1} f_{l,i,j,k}}{a_{l-1,i,j,k-1}} + \frac{f_{l-1,i,j,k} g_{l-1,i,j,k}}{a_{l-1,i,j,k}} \quad l \text{ mod } 4 = 2; \quad (4.32)
\]

\[
\left( \frac{q_l}{q_{l-1}} \right)^2 = \frac{b_{l,i,j,k-1} c_{l,i,j,k}}{a_{l-1,i,j,k-1}} + \frac{f_{l,i,j,k-1} b_{l,i,j,k}}{a_{l-1,i,j,k}} \quad l \text{ mod } 4 = 0. \quad (4.33)
\]

As is evident, Equations (4.31)-(4.33) are sufficient to determine all the diagonal entries, except the first entry in each $2n \times 2n$ block, which at this stage can be arbitrarily chosen. We have to make sure, then, that Eq. (4.27)-(4.30) are consistent with these three equations, and this requirement imposes some additional conditions. In the constant coefficient case there is unconditional consistency. The problematic nature of the variable coefficient case can be demonstrated simply by looking at one of the consistency conditions. Consider the $(i, j, k)$ gridpoint whose associated index, $l$, satisfies $l \text{ mod } 4 = 0$. Applying (4.31) to $l - 1$ means looking at the row corresponding to the $(i, j - 1, k - 1)$ gridpoint, and multiplying (4.31),
applied to \( l - 1 \), by (4.33) results in an equation for \( \frac{q_i}{q_{l-2}} \), which should be consistent with (4.30). The consistency condition is translated into the following:

\[
\begin{align*}
&\frac{g_{i-1,j-1,k-1} a_{i,j-1,k} + c_{i,j-1,k} f_{i,j-1,k}}{d_{i-1,j-1,k-1} f_{i-1,j-1,k} a_{i,j-1,k} + d_{i-1,j-1,k} f_{i,j-1,k} a_{i,j-1,k} - 1} \\
&\times \frac{b_{i,j,k-1} f_{i,j,k} a_{i,j-1,k} + b_{i,j,k} f_{i,j-1,k} a_{i,j,k-1}}{g_{i,j,k-1} e_{i,j-1,k} a_{i,j-1,k} + c_{i,j-1,k} g_{i,j-1,k-1} a_{i,j,k-1}} = \\
&\frac{b_{i,j,k} e_{i,j,k} d_{i,j,k} - 1}{d_{i-1,j,k} c_{i,j-1,k} a_{i,j-1,k} + c_{i,j-1,k} d_{i-1,j,k} a_{i,j,k-1}}.
\end{align*}
\]

(4.34)

There are three additional consistency conditions for the main block, and eight additional conditions for the rest of the blocks of the reduced matrix. Equating variables that belong to the same location in the computational molecule, necessary conditions for the above-mentioned consistency conditions to hold are \( b_{i,j,k} = b_{i,j,k-1}, \ c_{i,j,k-1} = c_{i,j,k}, \ d_{i,j,k-1} = d_{i,j,k}, \ e_{i,j,k-1} = e_{i,j,k} \). Under these conditions Eq. (4.34) becomes

\[
\frac{c_i g_{k-1}}{f_k d_{l-1}} \cdot \frac{b_j f_k}{c_{j-1} g_{k-1}} = \frac{b_j c_i}{d_{l-1} e_{j-1}},
\]

which is obviously satisfied.

The analysis for off-diagonal blocks is identical, and the following additional conditions are obtained:

\[
\left( \frac{q_l}{q_{l-2n}} \right)^2 = \frac{f_{i,j,k} f_{i,j,k-1}}{g_{i,j,k-1} g_{i,j,k-2}}, \quad 2n \leq l \leq \frac{n^3}{2};
\]

(4.36)

\[
\left( \frac{q_l}{q_{l-n^2}} \right)^2 = \frac{b_{i,j-1,k} b_{i,j,k}}{c_{i,j-1,k} c_{i,j-2,k}}, \quad n^2 \leq l \leq \frac{n^3}{2}.
\]

(4.37)

The above two equations determine the rest of the entries of the matrix, and only the first entry in the symmetrizer can be determined arbitrarily.

Last, in order for the symmetrizer to be real, the products \( c_i d_{l-1}, \ b_j e_{l-1} \) and \( f_k g_{k-1} \) must have the same sign. The actual meaning of the conditions stated above is that the
Chapter 4. Convergence Analysis for Block Stationary Methods

continuous problem has to be separable: in (2.11) the PDE variable coefficients should satisfy
\[ p = p(x), \quad q = q(y), \quad r = r(z), \quad s = s(x), \quad t = t(y) \quad \text{and} \quad w = w(z). \]
In this sense the three-dimensional problem has the same behavior as the two-dimensional problem [42]. We can now summarize all that has been said in the following symmetrization theorem:

**Theorem 4.3.** Suppose the operator of (2.11) is separable. If \( c_i d_{i-1}, b_j e_{j-1} \) and \( f_k g_{k-1} \) are all nonzero and have the same sign for all \( i, j \) and \( k \), then there exists a real nonsingular diagonal matrix \( Q \) such that \( Q^{-1} SQ \) is symmetric.

As in the constant coefficient case, the symmetrized computational molecule should be derived without actually performing the similarity transformation. For example, the symmetrized value corresponding to \(- \frac{c_i d_{i-1}}{a_i j, k} \) is \(- \frac{c_i d_{i-1} + c_i d_{i-1}}{a_i j, k} \); the symmetrized value corresponding to \(- \frac{c_i d_{i-1}}{a_i j, k} \) is \(- \frac{c_i d_{i-1} + c_i d_{i-1}}{a_i j, k} \), and so on. If the problem is separable the value at the center of the computational molecule is given by \( a_{i, j, k} = \frac{f_k g_{k-1}}{a_{i, j, k}} - \frac{c_i d_{i-1}}{a_{i, j, k}} \), and is unchanged under the similarity transformation.

4.2 Bounds on Convergence Rates

Below we attach the subscripts '1' and '2' to matrices associated with the 1D and 2D partitionings respectively. Let \( S = D_1 - C_1 = D_2 - C_2 \) be the corresponding splittings, and let \( \hat{D}_i = Q^{-1} D_i Q \) and \( \hat{C}_i = Q^{-1} C_i Q \), where \( Q \) is the diagonal symmetrizer. Below we refer to the particular matrix 2P2N\( n \times 2n \) (convergence results for other matrices in this family can be obtained by appropriate interchange of indices or roles of \( x, y \) and \( z \)). Consider the constant coefficient case, with \( bc, cd, fg > 0 \).

**Definition 4.1.** An interior block is a \( 2n \times 2n \) block \( \hat{S}^{(0)}_{i,j} \) whose associated set of grid points consists of points whose \( y \) and \( z \) coordinates are not \( 1/(n+1) \) or \( n/(n+1) \). No restriction is imposed on \( x \) coordinates.

We can focus on interior blocks, because the minimal eigenvalues of \( \hat{D}_i \) are sought; since non-interior blocks differ only on their diagonals, and these are larger algebraically than the
diagonals of interior blocks if \( be, \ cd, \ fg > 0 \), the latter have larger minimal eigenvalues, compared to interior blocks. Let us define a few auxiliary matrices and constants:

\[
\begin{align*}
    r &= -2 \sqrt{be \ fg} ; \ s = -2 \sqrt{cdfg} ; \ R_n = \text{tri}[E_{01}, 0, E_{10}] ; \\
    S_n &= \text{tri}[E_{10}, 0, E_{01}] ; \ T_n = \text{tri}[1, 0, 1] ; \ U_n = \text{penta}[1, 0, 0, 0, 1] ; \\
    V_n &= \text{septa}[E_{10}, 0, 0, 0, 0, E_{01}] ; \ Z_n = \text{tri}[s, r, s] .
\end{align*}
\]

The subscript \( n \) stands for the order of the matrices. Notice that all the above matrices are symmetric (see Sec. 1.3 for explanation of the notation). The matrix \( U_n^2 \) has 1’s on its fourth superdiagonal and subdiagonal, 2’s on its main diagonal except for the first two entries and last two entries where the values are 1, and zeros elsewhere. If we define

\[
W_n = (a^2 - 2be - 2fg) \cdot I_n - 2 \sqrt{bcede} \cdot U_n - cd \cdot U_n^2
\]

and

\[
X_n = -2 \sqrt{cdfg} \cdot (R_n + V_n) - 2 \sqrt{befg} \cdot S_n ,
\]

then an interior block of \( \hat{D}_1 \) is given by

\[
\hat{S}^{(0)}_{i,j} = W_{2n} + X_{2n} .
\]

We now examine the eigenvalues of \( W_{2n} \) and \( X_{2n} \). In the following, we shall make use of Kronecker products (see, for example, Barnett [10]), denoted by the symbol \( \otimes \). Recall that for any two matrices \( A \) and \( B \) of sizes, say, \( m \times n \) and \( p \times q \) respectively,

\[
A \otimes B = \begin{pmatrix}
    a_{1,1}B & a_{1,2}B & \cdots & a_{1,n}B \\
    \vdots & \vdots & & \vdots \\
    a_{m,1}B & a_{m,2}B & \cdots & a_{m,n}B
\end{pmatrix}
\]

In (4.42) each of the blocks is of size \( p \times q \), and \( A \otimes B \) is of size \( (mp) \times (nq) \). Kronecker products have a few useful properties [10], of which we mention in particular the ones which we will use for deriving bounds:

- Any four given matrices \( A, B, C, D \) with the appropriate sizes satisfy

\[
(A \otimes B)(C \otimes D) = (AC) \otimes (BD) .
\]
If \( \{\lambda_i, u_i\} \) and \( \{\mu_i, v_i\} \) are the sets of eigenvalues/eigenvectors of \( A \) and \( B \) respectively,
then
\[
(A \bigotimes B)(u_i \bigotimes v_i) = (\lambda_i \mu_i)(u_i \bigotimes v_i). \quad (4.44)
\]

Using these properties, we have:

**Lemma 4.1.** The eigenvalues of \( W_{2n} \) are given by
\[
a^2 - 2be - 2fg - 4\sqrt{bcde} \cdot \cos(\pi jh) - 4cd \cos^2(\pi jh), \quad j = 1, \ldots, n, \quad (4.45)
\]
each with algebraic multiplicity of 2.

**Proof.** Since
\[
U_{2n} = T_n \bigotimes I_2, \quad (4.46)
\]
this matrix's eigenvalues are \( \{2 \cos(\pi jh)\}_{j=1}^n \), each of algebraic multiplicity 2. \( W_n \) is a polynomial in \( U_n \), therefore it has the eigenvalues stated in (4.45).

**Lemma 4.2.** The matrix \( X_{2n} \) has the following eigenvalues:
\[
\lambda_j^\pm = \pm[2 \sqrt{bcfg} + 4 \sqrt{cd} fg \cdot \cos(\pi jh)]^\pm, \quad j = 1, \ldots, n. \quad (4.47)
\]

**Proof.** Suppose \( Y_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \). Then
\[
X_{2n} = Z_n \bigotimes Y_2, \quad (4.48)
\]
and thus by (4.44) the eigenvalues of \( X_{2n} \) are all the combinations of products of eigenvalues
of \( Z_n \) and \( Y_2 \), given by \( [r + 2s \cos(\pi jh)] \cdot (\pm 1), \ 1 \leq j \leq n. \)
Chapter 4. Convergence Analysis for Block Stationary Methods

Theorem 4.4. The eigenvalues of interior blocks \( S^{(0)}_{j,j} \) are

\[
p_j^\pm = a^2 - 2be - 2fg - 4\sqrt{be} \cdot \cos(\pi j h) - 4cd \cdot \cos^2(\pi j h)
\]

\[
\pm [2\sqrt{be}fg + 4\sqrt{cd}fg \cdot \cos(\pi j h)], \quad j = 1, \ldots, n.
\] (4.49)

Proof. Applying (4.43), using (4.46) and (4.48), we have

\[
U_{2n}X_{2n} = (T_n Z_n) \bigotimes (I_2 Y_2)
\] (4.50a)

\[
X_{2n} U_{2n} = (Z_n T_n) \bigotimes (Y_2 I_2)
\] (4.50b)

Since \( I_2 Y_2 = Y_2 I_2 = Y_2 \) and \( T_n Z_n = Z_n T_n \) we conclude that

\[
X_{2n} U_{2n} = U_{2n} X_{2n},
\] (4.51)

hence \( X_{2n} \) and \( U_{2n} \) commute, which means that \( X_{2n} \) and \( W_{2n} \) have common eigenvectors, can be simultaneously diagonalized, and the eigenvalues of \( S^{(0)}_{j,j} \) are the sum of the eigenvalues of \( X_{2n} \) and \( W_{2n} \), given in (4.49). \( \square \)

We remark that another way of analyzing the spectrum of \( X_{2n} \) is by using the relation \( (X_{2n})^2 = (Z_{2n})^2 \). We are now ready to prove

Theorem 4.5. The symmetric matrix \( \hat{D}_1 \) is positive definite if \( be, cd, fg > 0 \). The eigenvalues given in (4.49) are also eigenvalues of \( \hat{D}_1 \), each of multiplicity \( (n/2 - 2)^2 \). The rest of the eigenvalues of \( \hat{D}_1 \) are all in the interval \( [\min_j (\mu_j), \max_j (\mu_j) + be + cd + fg] \). The minimal eigenvalue of \( \hat{D}_1 \), \( \min_j (\mu_j) \), is given by

\[
\eta = a^2 - 2be - 2fg - 2\sqrt{be}fg - 4(\sqrt{be} + \sqrt{cd}) \cdot \cos(\pi h) - 4cd \cdot \cos^2(\pi h).
\] (4.52)

Proof. The eigenvalues of \( \hat{D}_1 \) are positive by combining Lemma 2.5, [113, Cor. 2] and [113, Thm. 3.12]. For the eigenvalues (4.49), the multiplicity is determined by counting the number of interior blocks. Since non-interior blocks differ from interior blocks only in their diagonal,
and only by 0 to \( bc + cd + fg \) in each diagonal, the minimal eigenvalue is attained in interior blocks and is thus given exactly by (4.52). Using equality between 2-norm and spectral radius of symmetric matrices, and the triangular inequality, results in an upper bound for the eigenvalues, given by \( \max_j (\mu_j) + bc + cd + fg \).

Next, \( \hat{D}_2 - \hat{D}_1 \) is considered. For a given value of \( j \) \((1 \leq j \leq \frac{n}{2})\), denote the \( j \)th block of \( \hat{D}_2 - \hat{D}_1 \) by \( S_{j,j} \). \( S_{j,j} = \text{tri}[\bar{S}_{j,j}^{(-1)}, 0, \bar{S}_{j,j}^{(-1)} T] \). \( S_{j,j} \) is an \( n^2 \times n^2 \) block tridiagonal matrix. All the block matrices \( \bar{S}_{j,j}^{(-1)} \), \( 1 \leq j \leq \frac{n}{2} \), are identical to each other, thus the analysis can be carried out for a single block.

Define \( Y_{2n}(r,s) \) to be the following \( 2n \times 2n \) matrix:

\[
Y_{2n}(r,s) = \begin{pmatrix}
0 & r & 0 & s \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & r & 0 & s \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & s & r & 0 \\
0 \\
0
\end{pmatrix}
\]

and let \( W(r,s) = \text{tri}[Y_{2n}(r,s), 0, Y_{2n}^T(r,s)] \) be an \( n^2 \times n^2 \) matrix, consisting of \( \frac{n}{2} \) blocks, each of size \( 2n \times 2n \). Then

\[
\bar{S}_{j,j} = \text{tri}[\bar{S}_{j,j}^{(-1)}, 0, \bar{S}_{j,j}^{(-1)} T] = \text{tri}[-fg I_{2n}, 0, -fg I_{2n}] + W(-2\sqrt{bcfg}, -2\sqrt{cdfg}) .
\]
Chapter 4. Convergence Analysis for Block Stationary Methods

Since $\tilde{S}_{j,j}$ is symmetric, its spectral radius can be estimated using

$$
\rho(\tilde{S}_{j,j}) = \|\text{tri}[-fg I_{2n}, 0, -fg I_{2n}] + W(-2\sqrt{b e f g}, -2\sqrt{c e d f g})\|_2 \\
\leq \|\text{tri}[-fg I_{2n}, 0, -fg I_{2n}]\|_2 + \|W(-2\sqrt{b e f g}, -2\sqrt{c e d f g})\|_2. \quad (4.55)
$$

The eigenvalues of the $n^2 \times n^2$ matrix $\text{tri}[-fg I_{2n}, 0, -fg I_{2n}]$ are $-2fg \cos\left(\frac{\pi j}{n}\right)$, $1 \leq j \leq \frac{n}{2}$, each with algebraic multiplicity of $2n$. Since it is a symmetric matrix, we obtain

$$
||\text{tri}[-fg I_{2n}, 0, -fg I_{2n}]||_2 = \rho(\text{tri}[-fg I_{2n}, 0, -fg I_{2n}]) = 2fg \cos\left(\frac{\pi}{2} + 1\right). \quad (4.56)
$$

To estimate $||W||_2$ in (4.55), it is easier to work with $W^2$, and use $||W||_2 = ||W^2||_2^{1/2}$. We have:

**Lemma 4.3.** The matrix $W^2$ has the following eigenvalues: 0, of multiplicity $2n$, and

$$
4befg + 16cdfg \cdot \cos^2(\pi jh) + 16\sqrt{bced} \cdot fg \cdot \cos(\pi jh), \quad 1 \leq j \leq n, \quad (4.57)
$$
each of multiplicity $n - 2$.

**Proof.** Forming $W^2$ in terms of $Y_{2n}$ and $Y_{2n}^T$, we get

$$
W^2 = \begin{pmatrix}
0 & (Y^T)^2 \\
0 & 0 & (Y^T)^2 \\
Y^2 & 0 & Y^T Y + YY^T & 0 & (Y^T)^2 \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
Y^2 & 0 & Y^T Y + YY^T & 0 & \cdots \\
Y^2 & 0 & YY^T & \cdots & \cdots \\
\end{pmatrix} \quad (4.58)
$$

[in Eq. (4.58) the subscripts $2n$ were dropped]. Now, a straightforward computation shows that $(Y_{2n})^2 = (Y_{2n}^T)^2 = 0$. From that it follows that $W^2$ is actually block diagonal.

The matrices $Y_{2n}Y_{2n}^T$ and $Y_{2n}^T Y_{2n}$, which appear in the first and last diagonal block entry
of $W^2$, are given by

\[
Y_{2n}Y_{2n}^T = \begin{pmatrix} r^2 + s^2 & 2rs & 0 & s^2 \\ 0 & 0 & 0 & 0 \\ 2rs & 0 & r^2 + 2s^2 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}
\]

(4.59)

\[
Y_{2n}^T Y_{2n} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & r^2 + s^2 & 0 & 2rs & 0 & s^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2rs & 0 & r^2 + 2s^2 & 0 & 2rs & 0 & s^2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}
\]

(4.60)

Combining (4.59) and (4.60), the matrix $Y_{2n}Y_{2n}^T + Y_{2n}^T Y_{2n}$, which appears in all the rest of the block diagonal entries of $W^2$, is given by
Chapter 4. Convergence Analysis for Block Stationary Methods

\[
\begin{pmatrix}
  r^2 + s^2 & 0 & 2rs & 0 & s^2 & 0 \\
  0 & r^2 + s^2 & 0 & 2rs & 0 & s^2 \\
  2rs & 0 & r^2 + 2s^2 & 0 & 2rs & 0 \\
  0 & 2rs & 0 & r^2 + 2s^2 & 0 & 2rs \\
  & \ddots & \ddots & \ddots & \ddots & \ddots \\
  & & \ddots & \ddots & \ddots & \ddots \\
  s^2 & 0 & 2rs & 0 & r^2 + 2s^2 & 0 \\
  s^2 & 0 & 2rs & 0 & r^2 + s^2 & 0 \\
  s^2 & 0 & 2rs & 0 & r^2 + s^2 \\
\end{pmatrix}
\]

(4.61)

From (4.61) it is evident that in terms of \(U_n\) [see Eq. (4.38)] we have

\[
Y_{2n}Y_{2n}^T + Y_{2n}^T Y_{2n} = r^2 \cdot I_{2n} + 2rs \cdot U_{2n} + s^2 \cdot U_{2n}^2,
\]

(4.62)

thus, the eigenvalues of this matrix, with \(r\) and \(s\) as in (4.38), are the ones given in Eq. (4.57). Each eigenvalue is of algebraic multiplicity 2.

The matrix \(Y_{2n}^T Y_{2n}\) can be permuted so that the \(n\) rows containing nonzero entries are first, followed by the zero rows. Doing so, we obtain a matrix of the type

\[
\begin{pmatrix}
  \bar{X} & 0 \\
  0 & 0 \\
\end{pmatrix}
\]

where \(\bar{X}\) is a symmetric pentadiagonal \(n \times n\) matrix given by penta\([s^2, 2rs, r^2 + 2s^2, 2rs, s^2]\), except the first and the last entries in the main diagonal are \(r^2 + s^2\).

Again, we use an auxiliary matrix that has been introduced in Eq. (4.38), and we have

\[
\bar{X} = r^2 \cdot I_n + 2rs \cdot T_n + s^2 \cdot T_n^2.
\]

(4.63)

From this it follows that the eigenvalues of \(Y_{2n}^T Y_{2n}\) and \(Y_{2n} Y_{2n}^T\) are thus exactly the ones given by (4.57), each of multiplicity 1, plus the eigenvalue 0, of multiplicity \(n\).

We can now find the eigenvalues of \(W^2\) by assembling all the eigenvalues of all blocks. Since
there are $\frac{\pi}{2} - 2$ blocks in $W^2$ that are equal to $Y_{2n}^T Y_{2n} + Y_{2n} Y_{2n}^T$, the result in the statement of the lemma follows.

Having the eigenvalues of $W^2$ at hand, we can now use equations (4.56) and (4.57) to obtain:

**Lemma 4.4.** The spectral radius of $\hat{D}_2 - \hat{D}_1$ can be bounded by

$$
\xi = 2fg \cos\left(\frac{\pi}{2} + 1\right) + \sqrt{4befg + 16 \cdot cdg \cdot \cos^2(\pi h) + 16 \sqrt{bcde \cdot fg \cdot \cos(\pi h)}}.
$$

(4.64)

Combining Thm. 4.5 and Lemma 4.4, and applying Rayleigh quotients to the matrices $\hat{D}_1$ and $\hat{D}_2 - \hat{D}_1$, we obtain:

**Lemma 4.5.** The minimal eigenvalue of $\hat{D}_2$ is bounded from below by $\eta - \xi$, where $\eta$ and $\xi$ are the expressions given in (4.52) and (4.64) respectively.

As a last step, we estimate the spectral radius of

$$
-\hat{C}_2 = \text{tr}\left[\hat{S}_{j,j-1}^{(-1)}, 0, \hat{S}_{j,j-1}^{(1) T}\right] + \text{tr}\left[\hat{S}_{j,j-1}^{(1)}, 0, \hat{S}_{j,j-1}^{(1)} T\right] + \text{tr}\left[\hat{S}_{j,j-1}^{(0)}, 0, \hat{S}_{j,j-1}^{(0)} T\right].
$$

(4.65)

**Lemma 4.6.** The spectral radius of $\text{tr}\left[\hat{S}_{j,j-1}^{(-1)}, 0, \hat{S}_{j,j-1}^{(-1) T}\right] + \text{tr}\left[\hat{S}_{j,j-1}^{(1)}, 0, \hat{S}_{j,j-1}^{(1)} T\right]$ is $2\sqrt{bcfg}$. Moreover, its eigenvalues are either 0, $+2\sqrt{bcfg}$ or $-2\sqrt{bcfg}$.

**Proof.** The square of the matrix is a diagonal matrix whose entries are either zeros or $4bcfg$. 

**Lemma 4.7.** The spectral radius of the $\frac{n^3}{2} \times \frac{n^3}{2}$ matrix $\text{tr}[be \cdot I_{n^2}, 0, -be \cdot I_{n^2}]$ is $2be \cdot \cos\left(\frac{\pi}{2} + 1\right)$.

**Proof.** Let $Z$ denote the $\frac{n}{2} \times \frac{n}{2}$ matrix $\text{tr}[1, 0, 1]$. Then $\text{tr}[be \cdot I_{n^2}, 0, -be \cdot I_{n^2}] = -be \cdot (Z \otimes I_{n^2})$. 

83
Chapter 4. Convergence Analysis for Block Stationary Methods

**Lemma 4.8.** The spectral radius of $\text{tri}\left[S_{\frac{3}{2}, \frac{3}{2}}^{(0)}, 0, S_{\frac{3}{2}, \frac{3}{2}}^{(0)}\right] - \text{tri}[-be \cdot I_{n^2}, 0, -be \cdot I_{n^2}]$ is given by

$$2 \sqrt{be fg} + 4 \sqrt{bcde} \cdot \cos(\pi h) \cdot (4.66)$$

**Proof.** Let $\tilde{C}_1$ denote the matrix that results from permuting the rows of the matrix given in the statement of the lemma so that indices whose moduli are 0 or 1 are indexed first, in increasing order, and indices whose moduli are 2 or 3 are indexed later. Let $\tilde{C}_2$ be a permutation of $\tilde{C}_1$ such that the rows and columns indexed $n^3/4 - n^2/2 + 1$ to $n^3/4 + n^2/2$ ($n^2$ such rows & columns) become rows/columns $n^3/2 - n^2 + 1$ to $n^3/2$ and the rest of the rows/columns are shifted accordingly. The last $n^2$ rows and columns of $\tilde{C}_2$ are zeros. If $\tilde{C}_3$ denotes the upper left $(n^3/2 - n^2) \times (n^3/2 - n^2)$ submatrix of $\tilde{C}_2$, then it is a block matrix of the form

$$\begin{pmatrix} 0 & \tilde{U} \\ \tilde{U} & 0 \end{pmatrix}$$

and $\tilde{U}$ is a block diagonal matrix consisting of $n \times n$ tridiagonal matrices given by $-2 \cdot \text{tri}[\sqrt{bcde}, \sqrt{be fg}, \sqrt{bcde}]$. In Fig. 4.1 the sparsity patterns of the above matrices are depicted.

![Sparsity patterns of matrices](image)

(a) $\tilde{C}_1$  (b) $\tilde{C}_2$  (c) $\tilde{C}_3$

Figure 4.1: Sparsity patterns of the matrices involved in the proof of Lemma 4.8

By Lemma 1.1 the spectral radius of $\hat{C}_3$ is the one given in (4.66). It is also the spectral radius of the matrix referred to in the statement of the lemma, as the rest of its eigenvalues are zero.

We can now combine the results obtained in Lemmas or Theorems 4.3 - 4.8 to obtain the following result:
Chapter 4. Convergence Analysis for Block Stationary Methods

Theorem 4.6. The spectral radius of \( \hat{C}_2 \) is bounded by

\[
\phi = 4\sqrt{b\epsilon f g + 4\sqrt{b\epsilon cde \cdot \cos\left(\frac{\pi}{n+1}\right)} + 2b\epsilon \cos\left(\frac{\pi}{n^2 + 1}\right)};
\]  

\( 4.67 \)

the spectral radius of \( \hat{C}_1 \) is bounded by \( \xi + \phi \) \( \xi \) is defined in Eq. \( 4.64 \)].

We are now ready to prove the main convergence result.

Theorem 4.7. The spectral radii of the iteration matrices \( D^{-1}_1C_1 \) and \( D^{-1}_2C_2 \) are bounded by \( \frac{\phi + \xi}{\eta} \) and \( \frac{\phi}{\eta - \xi} \) respectively, where \( \eta, \xi \) and \( \phi \) are defined in Eq. \( 4.52 \), \( 4.64 \) and \( 4.67 \), respectively.

From Theorem 4.7 we can draw the following conclusion with regard to the convergence of the block Jacobi scheme:

Corollary 4.2. If \( b, c, d, f, g > 0 \) then the block Jacobi iteration converges for both the 1D and 2D splittings.

Proof. The Taylor expansions of the bounds given in Theorem 4.7 are given by:

\[
\rho(D^{-1}_1C_1) \leq \frac{\phi + \xi}{\eta} = 1 - \left(\frac{10}{9}\pi^2 + \frac{1}{6}\mu^2 + \frac{1}{6}\tau^2 + \frac{1}{6}\sigma^2\right)h^2 + o(h^2)
\]  

\( 4.68 \)

and

\[
\rho(D^{-1}_2C_2) \leq \frac{\phi}{\eta - \xi} = 1 - (2\pi^2 + \frac{3}{10}\mu^2 + \frac{3}{10}\tau^2 + \frac{3}{10}\sigma^2)h^2 + o(h^2),
\]  

\( 4.69 \)

thus smaller than 1.

We remark that convergence of the scheme can also be deduced by [113, Thm. 3.13]. In Tables 4.2 and 4.1 we give some indication on the quality of the bounds. As can be observed, the bounds are tight and become tighter as \( n \) increases, which suggests that they tend to the actual spectral radii as \( h \to 0 \). Another important point is that the actual spectral radii are bounded far below 1.
Chapter 4. Convergence Analysis for Block Stationary Methods

<table>
<thead>
<tr>
<th>scheme</th>
<th>upwind</th>
<th>centered</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>$\rho$</td>
<td>bound</td>
</tr>
<tr>
<td>8</td>
<td>0.641</td>
<td>0.717</td>
</tr>
<tr>
<td>12</td>
<td>0.720</td>
<td>0.758</td>
</tr>
<tr>
<td>16</td>
<td>0.752</td>
<td>0.775</td>
</tr>
<tr>
<td>20</td>
<td>0.768</td>
<td>0.783</td>
</tr>
<tr>
<td>24</td>
<td>0.778</td>
<td>0.788</td>
</tr>
</tbody>
</table>

Table 4.1: comparison between the computed spectral radius and the bound, for the 1D splitting, with $\beta = 0.4, \gamma = 0.5, \delta = 0.6$.

<table>
<thead>
<tr>
<th>scheme</th>
<th>upwind</th>
<th>centered</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>$\rho$</td>
<td>bound</td>
</tr>
<tr>
<td>8</td>
<td>0.497</td>
<td>0.583</td>
</tr>
<tr>
<td>12</td>
<td>0.585</td>
<td>0.632</td>
</tr>
<tr>
<td>16</td>
<td>0.625</td>
<td>0.653</td>
</tr>
<tr>
<td>20</td>
<td>0.645</td>
<td>0.664</td>
</tr>
<tr>
<td>24</td>
<td>0.657</td>
<td>0.671</td>
</tr>
</tbody>
</table>

Table 4.2: comparison between the computed spectral radius and the bound, for the 2D splitting, with $\beta = 0.4, \gamma = 0.5, \delta = 0.6$.

For the variable coefficient case, recall from the previous section that if the problem is separable the matrix can be symmetrized by a real diagonal similarity transformation. Inspired by Elman & Golub’s strategy [42, Cor. 2], and the technique that is used to prove it, Theorem 4.7 can be generalized as follows:

**Theorem 4.8.** Suppose the continuous problem is separable, and $e_{i+1}d_i, b_{j+1}c_j$ and $f_{k+1}g_k$ are all positive and bounded by $\beta_x$, $\beta_y$ and $\beta_z$ respectively. Suppose also that $a_{i,j,k} \geq \alpha$ for all $i$, $j$ and $k$. Denote $\bar{h} = \frac{1}{\beta+1}$. Then the spectral radii of the iteration matrices associated with the block Jacobi scheme which correspond to 1D splitting and 2D splitting are bounded by $\frac{\phi + \xi}{\eta}$. 

86
Chapter 4. Convergence Analysis for Block Stationary Methods

\begin{equation}
\frac{\phi}{\eta - \xi} \text{ respectively, where:}
\end{equation}

\begin{align*}
\eta &= \alpha^2 - 2\beta_y - 2\beta_z - 2\sqrt{\beta_y\beta_z} - 4\sqrt{\beta_x}\cos(\pi h) - 4\beta_x \cos^2(\pi h) ; \\
\xi &= 2\beta_z \cos(\pi h) + \sqrt{4\beta_y\beta_z + 16\beta_x\beta_z \cos^2(\pi h)} + 16\beta_x \beta_y \cos(\pi h) ; \\
\phi &= 4\sqrt{\beta_y\beta_z} + 4\sqrt{\beta_x\beta_y} \cos(\pi h) + 2\beta_x \cos(\pi h) .
\end{align*}  

(4.70a)

(4.70b)

(4.70c)

**Proof.** The conditions stated in the theorem guarantee that the matrix is symmetrizable by a real diagonal nonsingular matrix. Denote the reduced matrix by $S$, and the symmetrized matrix by $\hat{S}$, and suppose $S^*$ is obtained by modifying the $\hat{S}$ in the following manner: replace each occurrence of $e_i$ and $d_i$ by $\sqrt{\beta_x}$, replace each occurrence of $b_j$ and $e_j$ by $\sqrt{\beta_y}$, replace each occurrence of $f_k$ and $g_k$ by $\sqrt{\beta_z}$ and replace each occurrence of $a_{i;j;k}$ by $a$. Denote by $S^* = D^* - C^*$ the splitting which is analogous (as far as sparsity pattern is concerned) to the splitting $S = D - C$. For the 1D splitting, the matrix $D^*$ is block diagonal with semibandwidth 4, its sparsity pattern is identical to that of $\hat{D}$, and it is componentwise smaller than or equal to the entries of $\hat{D}$. By Cor. 2.5, $D^*$ is a diagonally dominant $M$-matrix.

Clearly, $C^* \geq \hat{C} \geq 0$. Thus the Perron-Frobenius theorem [113, p. 30] can be used to obtain an upper bound on the convergence rate. Since the matrix $S^*$ can be now referred to as a symmetrized version of a matrix that is associated with a constant coefficient case, the bounds on the convergence rates are readily obtained from Thm. 4.7.

\[ \square \]

4.3 "Near-Property A" for 1D Partitioning of the Two-Plane Matrix

The block Jacobi scheme converges slowly, and thus is not satisfactory. We are interested in performing convergence analysis for the block SOR solver, which is typically very efficient if the optimal relaxation parameter (or a good approximation to it) is known. Considering the two-plane ordering, the reduced matrix is consistently ordered relative to the 2D partitioning
(by Prop. 3.3), and thus in this case Young’s analysis can be straightforwardly applied. On the other hand, analyzing the 1D partitioning is more challenging: by Prop. 3.2 the reduced matrix does not have Property A relative to this partitioning. However, in Section 4.4 it will be shown that the 1D partitioning gives rise to a more efficient solution process. It is thus important to provide some analysis for this case.

Our analysis and experimental observations are for the constant coefficient case and refer mainly to centered difference discretization. First, we consider the (easier) case of 2D partitioning and provide an approximation to the optimal relaxation parameter. We then consider the 1D partitioning: the spectral radius of the Jacobi iteration matrix is bounded in terms of spectral radii of two iteration matrices, one of which is associated with a consistently ordered matrix, and one whose spectral radius is small. We then use Varga’s extensions of the theory of $p$-cyclic matrices [112],[113, Sec. 4.4] (we are concerned with $p = 2$) to show that that relative to 1D partitioning, the reduced matrix behaves like a consistently ordered matrix.

We begin with the 2D partitioning. For this case we have the following:

**Theorem 4.9.** Let $L_{\omega}$ denote the block SOR operator associated with 2D splitting and using two-plane ordering. If $bc, cd, fg > 0$ or $< 0$ then the choice $\omega^* = \frac{2}{1 + \sqrt{1 - \rho^2(D_2^{-1}C_2)}}$ minimizes $\rho(L_{\omega})$ with respect to $\omega$, and $\rho(L_{\omega^*}) = \omega^* - 1$.

The proof of this theorem directly follows from [117, §5.2 and §14.3] and is essentially identical to the proof of [41, Thm. 4]. The algebraic details on how to pick the signs of the diagonal symmetrizer so that the symmetrized block diagonal part of the splitting is a diagonally dominant $M$-matrix are omitted. By Cor. 4.2 it is known that $\rho(D_2^{-1}C_2) < 1$. The reduced matrix is consistently ordered by Prop. 3.3.

A way to approximately determine the optimal relaxation parameter for $bc, cd, fg > 0$ is to replace $\rho(D_2^{-1}C_2)$ by the bound for it (given in Theorem 4.7) in the expression for $\omega^*$ in Theorem 4.9. If the bound for the block Jacobi scheme is tight, then the estimate of $\omega^*$ is fairly accurate.
Chapter 4. Convergence Analysis for Block Stationary Methods

**Corollary 4.3.** Suppose $bc, cd, fg > 0$. For the system associated with 2D splitting and for $h$ sufficiently small, the choice

$$\tilde{\omega}^* = \frac{2(\eta - \xi)}{\eta - \xi + \sqrt{(\eta - \xi)^2 - \tilde{\omega}^2}}$$

(4.71)

approximately minimizes $\rho(L_\omega)$. The spectral radius of the iteration matrix is approximately $\tilde{\omega}^* - 1$.

It is straightforward to verify that if $\tilde{\omega}^*$ is an accurate approximation to the optimal relaxation parameter, the asymptotic convergence rate of the SOR scheme is $O(h)$.

Moving to the 1D partitioning, as mentioned above, it is fundamentally different from the 2D partitioning and an analogous result to the one stated in Theorem 4.9 cannot be obtained by using Young’s analysis. However, for $bc, cd, fg > 0$ we have noticed that in many senses the reduced matrix does behave like a consistently ordered matrix relative to 1D partitioning.

For example, we have observed that the spectral radius of the block Jacobi iteration matrix satisfies $\rho_J^2 \approx \rho_{GS}$, and the graphs in Fig. 4.2 illustrate this phenomenon numerically. The broken lines correspond to the square of the spectral radius of the iteration matrix associated with block Jacobi, for an $8 \times 8 \times 8$ grid and $\beta = \gamma = \delta = 0.5$. The solid lines correspond to the spectral radius of the block Gauss-Seidel iteration matrix. As can be seen, the curves are almost indistinguishable. This phenomenon becomes more dramatic as the systems become larger.

![Graphs](image)

(a) $\rho_{GS}$ vs. $\rho_J^2$ - centered  (b) $\rho_{GS}$ vs. $\rho_J^2$ - upwind

Figure 4.2: “Near Property A” for the 1D splitting

Let $\{S_{i,j}\}$ denote the $n^2 \times n^2$ blocks of the reduced matrix. Each block $S_{i,j}$ is a block
Chapter 4. Convergence Analysis for Block Stationary Methods

tridiagonal matrix relative to $2n \times 2n$ blocks. Define

$$C_1^{(1)} = -\left( S_{j,j+1}^{(-1)} + S_{j,j+1}^{(1)} + S_{j,j-1}^{(-1)} + S_{j,j-1}^{(1)} \right), \tag{4.72}$$

where the matrices on the right hand side are defined in Sec. 3.2. As before, let $S = D_1 - C_1$ be the 1D Jacobi splitting of the matrix, and define $C_1^{(2)}$ so that $S = D_1 - (C_1^{(1)} + C_1^{(2)})$. The matrix $D_1 - C_1^{(2)}$ is consistently ordered, but $S$ is not, and does not have Property A either. It is convenient to consider the very sparse matrix $C_1^{(1)}$ as one which prevents $S$ from having Property A. In this matrix the magnitudes of the nonzero values are bounded by 2 if $be, cd, fg > 0$ and centered difference discretization is used for the convective terms. The nonzero pattern of $C_1^{(1)}$ is depicted in Fig. 4.3.

![Figure 4.3: sparsity pattern of the matrix C1^{(1)}.](image)

When $be, cd, fg > 0$ the reduced matrix $S$ is a diagonally dominant $M$-matrix (by Lemma 2.5), which can be symmetrized by a real diagonal similarity transformation (by Theorem 4.1). By Theorem 4.5 the symmetrized matrix $\hat{D}_1$ is positive definite, thus $\hat{D}_1^{1/2}$ is well defined. This leads to

**Lemma 4.9.** The spectral radius of the 1D Jacobi iteration matrix associated with the reduced matrix $S$ is bounded from above and below as follows:\footnote{I would like to thank Howard Elman for pointing out and providing the proof of the right-hand inequality in (4.73).}

$$\rho(D_1^{-1}C_1^{(2)}) \leq \rho(D_1^{-1}C_1^{(1)}) \leq \rho(D_1^{-1}C_1^{(2)}) + \rho(D_1^{-1}C_1^{(1)}). \tag{4.73}$$
Chapter 4. Convergence Analysis for Block Stationary Methods

Proof. Let the matrices $C_1^{(1)}$ and $C_1^{(2)}$ denote the symmetrized versions of $C_1^{(1)}$ and $C_1^{(2)}$ respectively. Since $\hat{D}_1^{-1}\hat{C}_1$ is similar to the symmetric matrix $D_1^{1/2}(\hat{D}_1^{-1}\hat{C}_1)\hat{D}_1^{-1/2} = \hat{D}_1^{-1/2}\hat{C}_1\hat{D}_1^{-1/2}$, we have

$$\rho(D_1^{-1}C_1) = \rho(\hat{D}_1^{-1}\hat{C}_1) = \rho(\hat{D}_1^{-1/2}\hat{C}_1\hat{D}_1^{-1/2}) = \|\hat{D}_1^{-1/2}\hat{C}_1\hat{D}_1^{-1/2}\|_2$$

$$\leq \|\hat{D}_1^{-1/2}\hat{C}_1^{(1)}\hat{D}_1^{-1/2}\|_2 + \|\hat{D}_1^{-1/2}\hat{C}_1^{(2)}\hat{D}_1^{-1/2}\|_2$$

$$= \rho(\hat{D}_1^{-1/2}\hat{C}_1^{(1)}\hat{D}_1^{-1/2}) + \rho(\hat{D}_1^{-1/2}\hat{C}_1^{(2)}\hat{D}_1^{-1/2}) = \rho(D_1^{-1}C_1^{(1)}) + \rho(D_1^{-1}C_1^{(2)}). \quad (4.74)$$

This completes the proof for the right-hand inequality. On the other hand, we have that both $C_1^{(2)}$ and $C_1$ are nonnegative matrices. Since $D_1$ is an $M$-matrix, and since $C_1^{(2)} \leq C_1^{(1)} + C_1^{(2)} = C_1$, we can use [95, Prop. 1.7], which states that if $A$, $B$ and $C$ are nonnegative matrices with $A \leq B$, then $CA \leq CB$ (and $AC \leq BC$). Hence $D_1^{-1}C_1^{(2)} \leq D_1^{-1}C_1$. Now, using [95, Thm. 1.14], which states that for two matrices $0 \leq A \leq B$ we have $\rho(A) \leq \rho(B)$ (this actually follows from the Perron-Frobenius theorem [113, p. 30]), the left-hand inequality in (4.73) follows. \hfill \Box

Table 4.3 illustrates numerically the inequalities given in Lemma 4.9 for a particular example. In the table, spectral radii of the iteration matrices defined above are given, for several cross-sections of mesh Reynolds numbers. The matrix for which the numbers are given is one arising from discretization on a $12 \times 12 \times 12$ grid. As is evident, the bounds are tight (by the analysis, the values in the second column of the table are bounded from above by the values in the third column, and from below by the values in the fourth column). In particular, the right-hand inequality is a very tight upper bound.

Another interesting point is that the spectral radius of the iteration matrix associated with $D_1 - C_1^{(1)}$ is fairly small. This can be quantified analytically, as follows:

Proposition 4.1. For $h$ sufficiently small, the spectral radius of the Jacobi iteration matrix associated with the splitting $D_1 - C_1^{(1)}$ is bounded from above by

$$\rho(D_1^{-1}C_1^{(1)}) \leq \frac{1}{9} - \left(\frac{7}{324}h^2 + \frac{7}{324}\tau^2 + \frac{4}{81}\pi^2 + \frac{1}{81}\sigma^2\right)h^2 + o(h^2). \quad (4.75)$$

91
Chapter 4. Convergence Analysis for Block Stationary Methods

<table>
<thead>
<tr>
<th>( \beta = \gamma = \delta )</th>
<th>( \rho(D_1^{-1}C_1^{(1)}) )</th>
<th>( \rho^{(1)} + \rho^{(2)} )</th>
<th>( \rho^{(2)} )</th>
<th>( \rho^{(1)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.8734</td>
<td>0.8762</td>
<td>0.7759</td>
<td>0.1002</td>
</tr>
<tr>
<td>0.3</td>
<td>0.7487</td>
<td>0.7511</td>
<td>0.6648</td>
<td>0.0862</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5441</td>
<td>0.5458</td>
<td>0.4827</td>
<td>0.0631</td>
</tr>
<tr>
<td>0.7</td>
<td>0.3151</td>
<td>0.3156</td>
<td>0.2788</td>
<td>0.0368</td>
</tr>
<tr>
<td>0.9</td>
<td>0.0979</td>
<td>0.0981</td>
<td>0.0866</td>
<td>0.0115</td>
</tr>
</tbody>
</table>

Table 4.3: comparison of computed spectral radii of the 1D Jacobi iteration matrix and matrices which are consistently ordered. The expressions in the header of the table stand for:

\[ \rho^{(1)} \equiv \rho(D_1^{-1}C_1^{(1)}) \quad \text{and} \quad \rho^{(2)} \equiv \rho(D_1^{-1}C_1^{(2)}). \]

Proof. In Lemma 4.6 the spectral radius of a matrix which is exactly equal to \(-\hat{C}_1^{(1)}\) is proved to be equal to \(2\sqrt{6\eta f g}\). Since \(\rho(D_1^{-1}C_1^{(1)}) = \rho(D_1^{-1}\hat{C}_1^{(1)}) \leq \frac{\rho(\hat{C}_1^{(1)})}{\lambda_{\min}(D_1)}\), it follows that \(\rho(D_1^{-1}C_1^{(1)}) \leq \frac{2\sqrt{6\eta f g}}{\eta}\), where \(\eta\) is as in Eq. (4.52). The leading terms of the Taylor expansion of this expression are given by Eq. (4.75).

We note that the matrix \(C_1^{(1)}\) has terms which are associated only with two independent variables \((y\) and \(z\) in this case), and thus the value in the Taylor expansion (4.75) which multiplies \(\tau^2\) and \(\mu^2\) (namely \(-\frac{1}{3274} h^2\)) is not equal to the value multiplying \(\sigma^2\).

The above analysis and experimental observations focus on the Jacobi iteration matrix. However, it is still not clear at this point whether the Gauss-Seidel (or SOR) eigenvalues behave in a way which resembles consistently ordered matrices. Here we can use Varga’s extensions of the theory of \(p\)-cyclic matrices [112], [113, Sec. 4.4]. Below we briefly describe some of Varga’s results, and use them for the reduced matrix. In [113, Defn. 4.2], a set \(V\) of matrices is defined as follows: \(^2\)

The square matrix \(B \in V\) if \(B\) satisfies the following properties:

\(^2\)In [113] the set is denoted by \(S\). Here, since \(S\) denotes the reduced matrix, the letter \(V\) is used.
1. \( B \geq 0 \) with zero diagonal entries.

2. \( B \) is irreducible and convergent.

3. \( B \) is symmetric.

Define \( \tilde{S} = \tilde{D}_1^{-1/2} \tilde{S} \tilde{D}_1^{-1/2} = I - \tilde{D}_1^{-1/2} \tilde{C}_1 \tilde{D}_1^{-1/2} \). Applying block Jacobi to the original reduced system is analogous to applying point Jacobi to \( \tilde{S} \), in the sense that the spectra of the iteration matrices associated with both systems are identical to each other. The iteration matrix associated with \( \tilde{S} \) is \( B = \tilde{D}_1^{-1/2} \tilde{C}_1 \tilde{D}_1^{-1/2} \). Showing that the matrix \( B \) belongs to the set \( V \) defined above is easy and is omitted. Let \( L \) be the lower part of \( B \). Define \( M_B(\theta) = \theta L + L^T, \theta \geq 0 \) and \( m_B(\theta) = \rho(M_B(\theta)) \). Let

\[
\hat{h}_B(\ln \theta) = \frac{m_B(\theta)}{\rho(B)^{\theta^{1/2}}}, \quad \theta > 0.
\] (4.76)

Then we have by [113, Theorem 4.7] that \( \hat{h}_B(\alpha) \equiv 1 \) if and only if \( B \) is consistently ordered. In some sense \( \hat{h}_B(\ln \theta) \) measures the departure of the matrix \( B \) from having block Property A. Fig. 4.4 demonstrates how close the function \( \hat{h}_B \) is to 1 for the reduced matrix when 1D partitioning is used, and is another way to illustrate the “near-Property A” of the matrix. In the figure, the function \( \hat{h}_B \) is computed for the same matrix for which Fig. 4.2(a) is given.

![Figure 4.4: the function \( \hat{h}_B(\alpha) \)](image)

Let \( \mathcal{L}_{B,\omega} \) denote the SOR iteration matrix. [113, Thm. 4.8] reads:

**Theorem 4.10.** If \( B \in V \) then

\[
\rho^2(B) \leq \rho(\mathcal{L}_{B,1}) < \frac{\rho(B)}{2 - \rho(B)},
\] (4.77)

with equality possible only if \( B \) is a consistently ordered cyclic matrix of index 2.
Chapter 4. Convergence Analysis for Block Stationary Methods

This is a sharpened form of the Stein-Rosenberg Theorem [113, p. 70]. Applying this theorem to our reduced matrix, we have:

**Theorem 4.11.** If the bound for the block Jacobi iteration matrix tends to the actual spectral radius as $h \to 0$, then the spectral radius of the block Gauss-Seidel iteration matrix coincides with the square of the bound for the spectral radius of the block Jacobi iteration matrix up to and including terms of order $h^2$.

**Proof.** Since the iteration matrix $B$ has the same spectral radius as $D_1^{-1}C_1$, we can use the bound of Thm. 4.7. For simplicity of notation, denote it by $\Phi$. Clearly, since $0 < \rho(B) \leq \Phi$,

$$\frac{\rho(B)}{2 - \rho(B)} \leq \frac{\Phi}{2 - \Phi} . \tag{4.78}$$

Since $\Phi$ has a Taylor expansion of the form $1 - ch^2 + o(h^2)$, by Eq. (4.68) it follows that $\frac{\Phi}{2 - \Phi}$ and $\Phi^2$ have the same Taylor expansion up to $O(h^2)$ terms, of the form $1 - 2ch^2 + o(h^2)$. Indeed, in terms of the PDE coefficients,

$$\frac{\Phi}{2 - \Phi} = 1 - \left( \frac{20}{9} \pi^2 + \frac{1}{3} \sigma^2 + \frac{1}{3} \tau^2 + \frac{1}{3} \mu^2 \right) h^2 + O(h^3) , \tag{4.79}$$

and the same for $\Phi^2$.

For the block SOR scheme, the upper bound for the spectral radius is given in [113, Theorem 4.9] as $\sqrt{\omega^* - 1}$ and is not tight. However, it is numerically evident that the bound for the Jacobi iteration matrix can be effectively used to estimate the optimal SOR parameter. In Fig. 4.5 we can observe that the behavior for the 1D splitting is qualitatively identical to the behavior of 2-cyclic consistently ordered matrices [the graph is given for the matrix that was used in Fig. 4.2(a)]. In the figure we also present the behavior of the SOR unreduc ed iteration matrix.

### 4.4 Computational Work

Having performed some convergence analysis, in this section we examine the question which of the solvers is more efficient overall. If $be, cd, fg > 0$, then by Eq. (4.68) and (4.69) (or by
Figure 4.5: spectral radius of the SOR iteration matrix vs. the relaxation parameter. The uppermost curve corresponds to 1D splitting for the unreduced matrix, and then, in order, 2D splitting for the unreduced matrix, 1D splitting for the reduced matrix, and 2D splitting for the reduced matrix.

[113, Thm. 3.15] it is evident that the spectral radius of the iteration matrix associated with the 2D splitting is smaller than that of the 1D iteration matrix. However, inverting $D_1$ involves less computational work than inverting $D_2$.

Consider the block Jacobi scheme. Asymptotically, in the constant coefficient case there is a fixed ratio of 1.8 between the rate of convergence of the two splittings - see Eq. (4.68) and (4.69). In rough terms, this number characterizes the ratio between number of iterations until convergence for the two solvers.

As far as the computational work per iteration is concerned, if $D_1 = L_1 U_1$ and $D_2 = L_2 U_2$ are the LU decompositions of the matrices of the systems that are to be solved in each iteration, we can assume that the number of operations per iteration is approximately the number of nonzeros in $L_i + U_i$ plus the number of nonzeros in $C_i$. In order to avoid costly fill-in using Gaussian elimination for $D_2$ (whose band is sparse), a technique of inner-outer iterations is used instead.

Let $k_1$ and $k_2$ denote the number of iterations for the schemes associated with the 1D splitting and the 2D splitting respectively. Let us also define cost functions, as follows: $c_1(n)$
and \( c_2(n) \) represent the overall number of floating point operations for each of the solvers, and \( c_{in}(n) \) represents the cost of the inner solve. Then

\[
c_1(n) \approx [n z(L_1 + U_1) + nz(S - D_1)] \cdot k_1 = [10n^3 - 19n^2 + 4n] \cdot k_1 ; \tag{4.80a}
\]

\[
c_2(n) \approx [c_{in}(n) + nz(S - D_2)] \cdot k_2 = [c_{in}(n) + 3n^3 - 8n^2 + 4n] \cdot k_2 . \tag{4.80b}
\]

In Eq. (4.80) the term \( n z(X) \) stands for the number of nonzeros of a matrix \( X \), and \( S \) stands for the reduced matrix.

**Proposition 4.2.** For \( n \) large enough, the scheme associated with the 2D splitting gives rise to a less costly solution process than the one associated with the 1D splitting only if \( c_{in}(n) < 15n^3 \).

**Proof.** If \( n \) is large enough we can use the relation \( \frac{k_1}{k_2} = 1.8 \) and refer only to the leading power of \( n \) in the expressions for \( c_1(n) \) and \( c_2(n) \). So doing, it follows that

\[
\frac{c_1(n)}{c_2(n)} \approx \frac{18n^3}{c_{in}(n) + 3n^3} , \tag{4.81}
\]

and the result stated in the proposition readily follows.

Next, the amount of work involved in solving the inner system of equations is determined. A natural choice of a splitting for this system is \( D_2 = D_1 - (D_1 - D_2) \). It is straightforward to show, by Lemma 4.4 and Theorem 4.5, that:

**Proposition 4.3.** If block Jacobi based on the splitting \( D_2 = D_1 - (D_1 - D_2) \) is used, then the spectral radius of the inner iteration matrix, namely \( I - D_1^{-1}D_2 \), is bounded by \( \frac{\xi}{\eta} \), where \( \eta \) and \( \xi \) are as in Eq. (4.52) and (4.64).

For considering stationary methods that are faster than block Jacobi for the inner system, we have:

**Proposition 4.4.** The inner matrix is block consistently ordered relative to 1D partitioning.

**Proof.** The inner matrix is block tridiagonal relative to this partitioning.
Chapter 4. Convergence Analysis for Block Stationary Methods

We now have:

**Theorem 4.12.** If be, cd, fg > 0, then for n large enough, and the stationary methods considered in this chapter, the 1D solver is faster than the 2D solver.

**Proof.** Below the result is shown for the block Jacobi outer solver; for Gauss-Seidel and SOR it is done in a similar way. The Taylor expansion of the bound in Prop. 4.3 is

\[
\frac{\xi}{\eta} = \frac{4}{9} - \left(\frac{43}{81} \pi^2 + \frac{65}{648} \mu^2 + \frac{29}{648} \tau^2 + \frac{25}{324} \sigma^2\right) h^2 + o(h^2). \tag{4.82}
\]

For \( h \) small enough, we can simply examine the leading term: the bound is approximately \( \frac{4}{9} \) if block Jacobi is used, and since by Prop. 4.4 the matrix is consistently ordered, by Young’s analysis the spectral radius is approximately \( \frac{16}{81} \) if block Gauss-Seidel is used, and approximately 0.055 if block SOR with the optimal relaxation parameter is used. For these schemes each iteration costs about \( 7n^3 \) floating point operations. Since reducing the initial error by a factor of \( 10^m \) takes roughly \( -m/\log_{10} \rho \) iterations, where \( \rho \) is the spectral radius of the associated iteration matrix, it follows that even for the block SOR scheme with the optimal relaxation parameter, which is the fastest scheme considered here, after two iterations the error is reduced only by a factor of approximately \( 10^{2.5} \) which is far from enough. Thus the iteration count is larger than two, and the cost of inner solve is larger than \( 15n^3 \) floating point operations. \( \square \)

We remark that an inexact inner solve could also be considered (see, for example, [43]), but we do not analyze this type of solver here.

### 4.5 Comparison with the Unreduced System

The unreduced matrix is an \( n^2 \)-order block tridiagonal matrix with respect to \( n^2 \times n^2 \) blocks:

\[
A = \text{tri}[A^{(-1)}, A^{(0)}, A^{(1)}],
\]

where \( A^{(0)} = f I_{n^2} \), \( A^{(1)} = g I_{n^2} \), and \( A^{(0)} \) are themselves block tridiagonal matrices,

\[
A^{(0)} = \text{tri}[B^{(-1)}, B^{(0)}, B^{(1)}],
\]

where
consisting of $n \times n$ matrices given by:

\[ B^{(-1)} = b \, I_n ; \quad B^{(0)} = \text{tridiag}[c, a, d] ; \quad B^{(1)} = e \, I_n . \] (4.85)

Consider the one-dimensional splitting

\[ A = D - C \] where

\[ D = \text{diag}[B^{(0)}, \ldots, B^{(0)}] . \] (4.86)

**Theorem 4.13.** If $cd > 0$, $be > 0$, and $fg > 0$ then there exists a real nonsingular diagonal matrix $Q$ such that the matrix $\hat{A} = Q^{-1}AQ$ is symmetric.

**Proof.** This can be shown by direct substitution, by requiring that the symmetrized matrix be equal to its transpose. Denote the $i$th entry in the main diagonal of $Q$ by $q_i$. $q_1 \neq 0$ can be an arbitrary value, and the following algorithm generates the desired diagonal matrix:

\begin{align*}
\text{for } i & = 2 \text{ to } n \\
q_i & = \sqrt{\frac{2}{a} \cdot q_{i-1}} \\
\end{align*}

\begin{align*}
\text{for } \ell & = 2 \text{ to } n \\
& \quad \text{for } i = 1 \text{ to } n \\
q_{(\ell-1)n+i} & = \sqrt{\frac{2}{e} \cdot q_{(\ell-2)n+i}} \\
\end{align*}

\begin{align*}
\text{for } j & = 2 \text{ to } n \\
& \quad \text{for } \ell = 1 \text{ to } n \\
& \quad \quad \text{for } i = 1 \text{ to } n \\
q_{(j-1)n^2+(\ell-1)n+i} & = \sqrt{\frac{2}{g} \cdot q_{(j-2)n^2+(\ell-1)n+i}} \\
\end{align*}

From the above it is clear that the similarity transformation is real and nonsingular only if $cd$, $fg$ and $be$ are positive. \hfill \square

98
Chapter 4. Convergence Analysis for Block Stationary Methods

The resulting symmetrized matrix is

\[ \hat{A} = \text{tri}[\hat{A}^{(-1)}, \hat{A}^{(0)}, \hat{A}^{(1)}], \]

where

\[ \hat{A}^{(-1)} = \hat{A}^{(1)} = \sqrt{fg} I_n; \quad \hat{A}^{(0)} = \text{tri}[\hat{B}^{(-1)}, \hat{B}^{(0)}, \hat{B}^{(1)}], \]

with

\[ \hat{B}^{(-1)} = \hat{B}^{(1)} = \sqrt{be} I_n; \quad \hat{B}^{(0)} = \text{tri}[\sqrt{cd}, a, \sqrt{cd}]. \]

Theorem 4.13 leads to the following symmetrization result, obtained by expressing the conditions in terms of \( \beta, \gamma \) and \( \delta \):

**Corollary 4.4.** If \( |\beta|, |\gamma|, |\delta| < 1 \), the coefficient matrix for the centered difference scheme is symmetrizable by a real diagonal similarity transformation. For upwind (backward) schemes, the coefficient matrix is symmetrizable for all \( \beta, \gamma, \delta > 0 \).

Cor. 4.4 demonstrates the similarity of the symmetrization conditions between the 2D [40] and 3D problems: in both cases the unreduced matrix can be symmetrized by a diagonal matrix only in the diffusion dominated case if centered differences are used, and in both cases the reduced matrix can be symmetrized for a larger range of PDE coefficients, including the case where all mesh Reynolds numbers are larger than 1 in magnitude.

We now find the spectrum of the iteration matrix. Let \( \hat{D} \) and \( \hat{C} \) be the symmetrized versions of \( D \) and \( C \) respectively. Then:

**Theorem 4.14.** The eigenvalues of the iteration matrix \( \hat{D}^{-1}\hat{C} \) are given by

\[
\frac{2\sqrt{fg} \cos(\pi jh) + 2\sqrt{be} \cos(\pi kh)}{a + 2\sqrt{cd} \cos(\pi \ell h)}, \quad 1 \leq j, k, \ell \leq n .
\]

**Proof.** Inspired by the techniques used by Elman & Golub in [40, p. 677], suppose \( V_n \) is an \( n \times n \) matrix whose columns are the eigenvalues of the tridiagonal matrix \( \text{tri}[\sqrt{cd}, a, \sqrt{cd}] \), and
let \( V = \text{diag}[V_n, \ldots, V_n] \) have \( n^2 \) copies of \( V_n \). Then \( \hat{A} = V^{-1} \hat{A} V \) diagonalizes \( \hat{D} \). Denote 
\( \tilde{D} = V^{-1} \hat{D} V \) and \( \tilde{C} = V^{-1} \hat{C} V \). Since \( V^{-1} \hat{D}^{-1} \hat{C} V = \tilde{D}^{-1} \tilde{C} \) we can find the spectrum of \( \tilde{D}^{-1} \tilde{C} \) by examining \( \tilde{D}^{-1} \tilde{C} \). The latter is easier to form explicitly, as \( \tilde{D} \) is diagonal (as opposed to \( \hat{D} \), which is tridiagonal). \( \tilde{D}^{-1} \tilde{C} \) has the same nonzero pattern of \( \tilde{C} \).

Let \( P_1 \) denote the permutation matrix that transforms rowwise ordering into columnwise ordering, leaving the ordering of planes (z direction) unchanged. \( F_1 = P_1^T \tilde{D}^{-1} \tilde{C} P_1 \) is block tridiagonal with respect to \( n^2 \times n^2 \) blocks, and its superdiagonal and subdiagonal blocks are diagonal matrices, all equal to each other. Each of these \( n^2 \times n^2 \) matrices looks as follows:

\[
\text{diag}\left[ \frac{\sqrt{fg}}{a + 2 \sqrt{cd} \cos(\pi h)}, \frac{\sqrt{fg}}{a + 2 \sqrt{cd} \cos(2\pi h)}, \ldots, \frac{\sqrt{fg}}{a + 2 \sqrt{cd} \cos(n\pi h)} \right].
\] (4.91)

The diagonal block consists of \( n \) identical \( n^2 \times n^2 \) identical matrices, each being an \( nth \)-order block diagonal matrix whose \( jth \) component is the tridiagonal matrix

\[
G_j = \text{tri}\left[ \frac{\sqrt{be}}{a + 2 \sqrt{cd} \cos(\pi j h)} \right].
\] (4.92)

Let \( V_{n^2} \) denote a matrix whose columns are the eigenvectors of \( \text{diag}[G_1, \ldots, G_n] \), and let \( V_2 \) be the block diagonal matrix consisting of \( n \) uncoupled copies of \( V_{n^2} \); then \( F_2 = V_2^{-1} F_1 V_2 \) is still block tridiagonal with respect to \( n^2 \times n^2 \) blocks, but now the main diagonal block is a diagonal matrix: it contains \( n \) identical \( n^2 \times n^2 \) blocks, each being a diagonal matrix whose entries are \( \frac{2 \sqrt{be} \cos(\pi j h)}{a + 2 \sqrt{cd} \cos(\pi k h)} \), \( 1 \leq j, k \leq n \). Finally, let \( P_2 \) be a permutation matrix which transforms rowwise ordering to planewise ordering, leaving the orientation of columns unchanged. Then \( F_3 = P_2^{-1} F_2 P_2 \) is an \( n^2 \)-order block diagonal matrix whose components are \( n \times n \) symmetric tridiagonal matrices. Their eigenvalues can be found using Lemma 1.1, and are given by (4.90). \(\)

Theorem 4.14 leads to the following useful result:

**Corollary 4.5.** For \( cd, be, fg > 0 \) the spectral radius of the line Jacobi iteration matrix for the unreduced system, using the preconditioner defined in (4.86), is

\[
\frac{2 \sqrt{be} \cdot \cos(\pi h) + 2 \sqrt{fg} \cdot \cos(\pi h)}{a + 2 \sqrt{cd} \cdot \cos(n\pi h)}.
\] (4.93)
Chapter 4. Convergence Analysis for Block Stationary Methods

The Taylor expansion of Eq. (4.90) about $h = 0$ is given by

$$1 - \left( \frac{3}{4} \pi^2 + \frac{1}{16} \sigma^2 + \frac{1}{16} \tau^2 + \frac{1}{16} \mu^2 \right) h^2 + o(h^2).$$  \hspace{1cm} (4.94)

For the plane Jacobi a similar procedure to the one used in the proof of Thm. 4.14 can be applied. (The algebraic details are omitted.)

**Theorem 4.15.** The spectral radius of the plane Jacobi iteration matrix for the unreduced system is given by

$$\frac{2 \sqrt{fg} \cdot \cos(\pi h)}{a + 2 \sqrt{cd} \cdot \cos(\pi nh) + 2 \sqrt{be} \cdot \cos(\pi nh)},$$  \hspace{1cm} (4.95)

and its Taylor expansion about $h = 0$ is

$$1 - \left( \frac{3}{2} \pi^2 + \frac{1}{8} \sigma^2 + \frac{1}{8} \tau^2 + \frac{1}{8} \mu^2 \right) h^2 + o(h^2).$$  \hspace{1cm} (4.96)

The same type of analysis that has been done in Section 4.4, comparing the 1D splitting to the 2D splitting for the reduced system, is possible for the unreduced system. Below we sketch the main details: suppose inner-outer iterations are used in solving the scheme associated with the 2D splitting. Denote, again, this splitting for the inner system as $D_2 = D_1 - (D_1 - D_2)$ ($D_1$ and $D_2$ are now different than the ones defined in the previous section). Then we have:

**Proposition 4.5.** Consider the unreduced system. Suppose $be, cd, fg > 0, n$ is sufficiently large, and 1D splitting is used in solving the inner system. Then for the stationary methods considered in this chapter, the 1D solver is faster than the 2D solver.

**Proof.** The ratio between the asymptotic rate of convergence between the 1D solver and the 2D solver is 2. The number of nonzeros of the whole matrix is approximately $7n^3$, the number of nonzeros of $D_1$ is approximately $3n^3$ and the number of nonzeros of $D_2$ is approximately $5n^3$. Since the spectral radii for the two splittings are available, we can find the spectral radius for the iteration matrix of the inner system. Its Taylor expansion is given by $\frac{1}{2} - \left( \frac{3}{8} \pi^2 + \frac{1}{16} \sigma^2 + \frac{1}{32} \tau^2 \right) h^2 + o(h^2)$. Defining cost functions analogous to the ones defined in Section 4.4 for the reduced system, and using the same line of argument, we have

$$\frac{c_1(n)}{c_2(n)} \approx \frac{14n^3}{c_{in}(n) + 2n^3},$$  \hspace{1cm} (4.97)

101
and from this it follows that only if $c_{in}(n) < 12n^3$ the 2D solver is more efficient. However, as in Thm. 4.12, this means that at most two iterations of the inner solve can be performed, which is not enough for the required accuracy.

Since the 1D splitting for both the reduced and the unreduced systems gives rise to a more efficient solve, we compare these two systems, focusing on this splitting. The LU decomposition for the solution of the system in each iteration is done once and for all (see [54] for operation count) and its cost is negligible in comparison with the amount of work done in the iterative process. Each iteration in the reduced system costs about $10n^3$ floating point operations whereas each iteration for the unreduced system costs approximately $7n^3$ floating point operations per iteration. Hence, the amount of computational work per iteration is cheaper for the unreduced system by a factor of about 10/7. However, using the asymptotic formulas (4.68) and (4.94), it is evident that the number of iterations required for the unreduced system is larger than that required for the reduced system, and in the worst case, the ratio between the work required for solving the reduced system vs. the unreduced system is roughly $(10/7) \cdot (27/40)$, which is 27/28 and is still smaller than 1, thus the reduced solver is more efficient. If the convective terms are nonzero, then this ratio becomes smaller, and in practice we have observed substantial savings, as is illustrated in the test problems in Sec. 4.7.

In Fig. 4.5 the superiority of the reduced system over the unreduced system for the Gauss-Seidel scheme is illustrated. The graphs were created for a 512-point grid. It is interesting to notice that the reduced Gauss-Seidel iteration matrices for both 1D and 2D splittings have spectral radii that are significantly smaller than 1 even for the convection-dominated case, when centered difference discretization is applied. The 2D splitting gives rise to a matrix with Property A, and in all the numerical experiments that have been performed for this splitting and for Gauss-Seidel and SOR the rate of convergence was much faster than for the unreduced system; in several cases the solver for the unreduced system did not converge. These observations are similar to the results in [107] for lower dimensions.
Chapter 4. Convergence Analysis for Block Stationary Methods

The superiority of the reduced system is evident also for the SOR scheme - see Fig. 4.5. Notice that for the SOR scheme it is difficult to determine the optimal relaxation parameter when \( be, cd \) and \( fg \) are negative.

![Comparison of spectral radii](image)

**Figure 4.6:** Comparison of the spectral radii of the Gauss-Seidel iteration matrices of the reduced and unreduced systems. Axes are: \( x \) - mesh Reynolds numbers \((\beta = \gamma = \delta)\); \( y \) - spectral radius. The uppermost curve corresponds to 1D splitting for the unreduced system, and then, in order, 2D splitting for the unreduced system, 1D splitting for the reduced system, and 2D splitting for the reduced system.

4.6 Fourier Analysis

We now perform Fourier analysis in the spirit of Chan & Elman [21] and Elman & Golub [40], for convection dominated equations: We are considering here centered difference discretization of convection-dominated equations where \( be, cd, fg < 0 \).

The boundary conditions are assumed to be periodic and the equation has constant coefficients. For the results presented below, we denote the periodic symmetrized reduced operator, scaled by \( ah^2 \), by \( \hat{S} = \hat{D} - \hat{C} \), where the question which matrices \( \hat{D} \) and \( \hat{C} \) are is to be clear in the context where they appear.

**Theorem 4.16.** Given \( be, cd, fg < 0 \), suppose \( 8\sqrt{|fg|} \cdot (\sqrt{|cd|} + \sqrt{|be|}) < a^2 \). If the two-line ordering with \( x-y \) oriented 2D blocks of gridpoints is used, then the eigenvalues of \( \hat{D}^{-1}\hat{C} \)
Chapter 4. Convergence Analysis for Block Stationary Methods

associated with the symmetrized periodic operator using 2D preconditioning are all smaller than 1 in magnitude.

Proof. Suppose \( v^{(\hat{\alpha}, \hat{\beta}, \hat{\gamma})} \) are vectors whose components are \( v_{j,k,l}^{(\hat{\alpha}, \hat{\beta}, \hat{\gamma})} = e^{2\pi i j \hat{\alpha} h} e^{2\pi i k \hat{\beta} h} e^{2\pi i l \hat{\gamma} h} \). Here \( i = \sqrt{-1} \), and \( 1 \leq j, k, l, \hat{\alpha}, \hat{\beta}, \hat{\gamma} \leq n \). For a grid point \( u_{i,j,k} \) we have

\[
\hat{D} u_{i,j,k} = (a^2 - 2be - 2cd - 2fg) u_{i,j,k} - be u_{i,j+2,k} - 2\sqrt{bcde} u_{i+1,j+1,k} - cd u_{i-2,j,k}
\]

\[
- cd u_{i+2,j,k} - 2\sqrt{bcde} u_{i-1,j-1,k} - be u_{i,j-2,k} - 2\sqrt{bcde} u_{i-1,j+1,k} - 2\sqrt{bcde} u_{i+1,j-1,k} \quad (4.98)
\]

and

\[
\hat{C} u_{i,j,k} = -fg u_{i,j,k-2} - 2\sqrt{bcfg} u_{i,j+1,k-1} - 2\sqrt{cfg} u_{i-1,j,k-1} - 2\sqrt{befg} u_{i,j-1,k-1} - 2\sqrt{cfg} u_{i,j+1,k+1}
\]

\[
- 2\sqrt{bcfg} u_{i-1,j,k+1} - 2\sqrt{befg} u_{i,j-1,k+1} - 2\sqrt{fg} u_{i,j-1,k+1} - fg u_{i,j,k+2} \quad (4.99)
\]

Substituting the eigenvectors presented above, by some algebra we get that the eigenvalues of \( \hat{D} \) are

\[
\lambda_{\hat{\alpha}, \hat{\beta}, \hat{\gamma}} = a^2 - 2be - 2cd - 2fg - 4\sqrt{bcde} \cdot [\cos(\theta + \phi) + \cos(\theta - \phi)]
\]

\[
- 2be \cos 2\phi - 2cd \cos 2\theta , \quad (4.100)
\]

where \( \theta = 2\pi \hat{\alpha} h, \phi = 2\pi \hat{\beta} h, \) and \( \xi = 2\pi \hat{\gamma} h \). The eigenvalues of \( \hat{C} \) are given by

\[
\mu_{\hat{\alpha}, \hat{\beta}, \hat{\gamma}} = 4\sqrt{bcfg} \cdot [\cos(\phi + \xi) + \cos(\phi - \xi)]
\]

\[
+ 4\sqrt{bcfg} \cdot [\cos(\theta + \xi) + \cos(\theta - \xi)] + 2fg \cos 2\xi . \quad (4.101)
\]

The eigenvalues of \( \hat{D}^{-1} \hat{C} \) are all the combinations of \( \mu/\lambda \). Denote \( p = \sqrt{|cd|} \), \( q = \sqrt{|be|}, \) \( r = \sqrt{|fg|} \). Note that \( cd = -p^2, \) \( be = -q^2, \) \( fg = -r^2 \). Then

\[
\frac{\mu}{\lambda} = \frac{8pr \cos \theta \cos \xi + 8qr \cos \phi \cos \xi - 4r^2 \cos^2 \xi + 2r^2}{a^2 + 2r^2 - 8pq \cos \theta \cos \phi + 4p^2 \cos^2 \theta + 4q^2 \cos^2 \phi} . \quad (4.102)
\]

For the denominator in (4.102) we have

\[
\lambda = a^2 + 2r^2 + 4(p \cos \theta - q \cos \phi)^2 \geq a^2 + 2r^2 . \quad (4.103)
\]
Chapter 4. Convergence Analysis for Block Stationary Methods

\[ \cos \xi \text{ appears only in the numerator, which is a quadratic function in it:} \]

\[ \mu(\cos \xi) = 2r^2(1 + s \cos \xi - 2\cos^2 \xi) \] \hspace{1cm} (4.104)

where

\[ s = 4 \cdot \frac{p \cos \theta + q \cos \phi}{r} \] \hspace{1cm} (4.105)

Let \( s_{\text{max}} = \max |s| = 4(p+q)/r \). If \( p+q \leq r \) then \( \mu \) has a local maximum at \(-1 \leq \cos \xi = \frac{1}{4} \leq 1\), and then

\[ \frac{\mu_{\text{max}}}{2r^2} = 1 + s^2/8 \leq 1 + s_{\text{max}}. \] \hspace{1cm} (4.106)

The minimal value of \( \frac{\mu}{2r^2} \) occurs at either \( \cos \xi = 1 \) or \( \cos \xi = -1 \), where its value is

\[ \frac{\mu_{\text{min}}}{2r^2} = -1 \pm s, \] \hspace{1cm} (4.107)

thus \( \mu_{\text{min}}/(2r^2) = -1 - |s| \) and from this it follows that the maximal absolute value of \( \mu/(2r^2) \) is bounded by \( 1 + |s| \). We conclude then, that for this case \( |\mu/(2r^2)| \leq 1 + s_{\text{max}}. \)

It now remains to check what happens if \( p+q > r \). In this case there are no local extrema for \( \mu \), and we have to look at the end points. Substituting \( \cos \xi = \pm 1 \), as above, we have \( \mu/(2r^2) = -1 \pm s \) which yields \( |\mu/(2r^2)| \leq 1 + s_{\text{max}}. \) We thus have

\[ |\mu| \leq 2r^2 \cdot \left[1 + \frac{4(p+q)}{r}\right]. \] \hspace{1cm} (4.108)

From (4.103) and (4.108) it follows that \( |\mu/\lambda| < 1 \) if \( 8r(p+q) < a^2 \), which in terms of the components of the computational molecule is exactly the condition stated in the theorem. \( \square \)

For the two-plane ordering, since the matrix entries were given in the previous section for 2PNxz, we refer to this particular ordering strategy below, and we have the following:

**Theorem 4.17.** Given \( be, cd, fg < 0 \), suppose \( 4\sqrt{|be|} \cdot (2\sqrt{|cd|} + \sqrt{|be|} + 2\sqrt{|fg|}) < a^2 \). Then the eigenvalues of \( \hat{D}^{-1}\hat{C} \) associated with the periodic operator using 2D preconditioning for the two-plane ordering are all smaller than 1 in magnitude.
Chapter 4. Convergence Analysis for Block Stationary Methods

Proof. For a grid point \( u_{i,j,k} \), the structure of the \( \hat{D} \) and \( \hat{C} \) depends on the parity of the index \( j \), associated with the independent variable \( y \). If \( j \) is even

\[
\hat{D} u_{i,j,k} = (a^2 - 2be - 2cd - 2fg) u_{i,j,k} - fg u_{i,j,k-2}
\]

\[-2\sqrt{cdfg} u_{i-1,j,k-1} - 2\sqrt{cdfg} u_{i+1,j,k-1} - 2\sqrt{befg} u_{i,j-1,k-1} - cd u_{i-2,j,k} - 2\sqrt{be} u_{i+1,j,k-1} - 2\sqrt{bede} u_{i+1,j-1,k} - 2\sqrt{be} u_{i+1,j,k-1} - 2\sqrt{be} u_{i+1,j,k+1} - 2\sqrt{be} u_{i,j,k+2}
\]

and

\[
\hat{C} u_{i,j,k} = -2\sqrt{be} u_{i,j+1,k-1} - be u_{i,j+2,k} - 2\sqrt{be} u_{i+1,j+1,k} - be u_{i,j-2,k} - 2\sqrt{be} u_{i+1,j+1,k} - 2\sqrt{be} u_{i-1,j+1,k}.
\] (4.109)

If \( j \) is odd, then the roles of \( j+1 \) and \( j-1 \) in \( \hat{D} \) and \( \hat{C} \) are interchanged. In both cases the eigenvalues are identical, and after some algebra we get:

\[
\frac{\mu}{\lambda} = \frac{4qr \cos \phi \cos \xi + 4pq \cos \phi \cos \phi + 2q^2 \cos 2\phi}{a^2 + 2q^2 - 8pr \cos \theta \cos \xi + 4p^2 \cos^2 \theta + 4r^2 \cos^2 \xi - 4qr \cos \phi \cos \xi - 4pq \cos \theta \cos \phi}.
\] (4.111)

We can write \( \lambda = a^2 + 2q^2 - 4(p \cos \theta - r \cos \xi)^2 - 4q \cos \phi (p \cos \theta + r \cos \xi) \), and if \( s = 2 \cdot \frac{p \cos \theta + r \cos \xi}{q} \), \( s_{\text{max}} = 2(p + q)/r \), we have

\[
\lambda \geq a^2 + 2q^2 - 2s_{\text{max}}q^2.
\] (4.112)

For the numerator, \( \mu \), we use a technique similar to the one used in Theorem 4.16: \( \mu \) is quadratic in \( \cos \phi \), and we have

\[
\mu = 2q^2(-1 + s \cos \phi - 2 \cos^2 \phi).
\] (4.113)

If \( |s| \leq 4 \) then \( \frac{\mu}{2q^2} \) obtains its maximum at \( \cos \phi = \frac{1}{4} \), where \( \mu/(2q^2) = -1 + (s^2)/8 \leq 1 \). The minimum is obtained at \( \cos \phi = \pm 1 \) and we have

\[
\mu_{\text{min}}/(2q^2) = -3 - |s|.
\] (4.114)

106
So, \[|\mu/(2q^2)| \leq 3 + s_{\text{max}}.\] Hence, it follows that
\[
\frac{\mu}{\lambda} \leq \frac{2(3 + s_{\text{max}})q^2}{a^2 + 2q^2 - 2s_{\text{max}}q^2}. \tag{4.115}
\]
The latter is smaller than 1 if \(4q(2p + q + 2r) < a^2.\)

### 4.7 Comparisons

In this section some indication as to the performance of block stationary methods for the reduced system is given, and the effectiveness of the cyclic reduction step is illustrated.

**Comparison between the reduced and the unreduced systems**

Consider Eq. (1.15) where the right hand side is such that the solution for the continuous problem is \(u(x, y, z) = \sin(\pi x) \cdot \sin(\pi y) \cdot \sin(\pi z),\) and the domain is the unit cube. The zero vector was taken as an initial guess, and \(\|r^{(i)}\|_2/\|r^{(0)}\|_2 < 10^{-10}\) was used as a stopping criterion (\(r^{(i)}\) denoting, as usual, the residual at the \(i\)th iterate). The program stopped if the stopping criterion was not satisfied after 2,000 iterations.

In the experiments that are presented, the 1D solver is used. In Table 4.4, the grid is of size \(32 \times 32 \times 32.\) The matrix of the underlying system of equations is of size \(32,768 \times 32,768.\) In the table, iteration counts for the Jacobi, Gauss-Seidel and SOR schemes are presented for four values of the PDE coefficients, and for two discretization schemes. It is straightforward to make the connection between iterations and computational work: since the problem has constant coefficients, the construction of both the reduced and the unreduced systems is straightforward and inexpensive, and is negligible in the overall cost of the computation (see Sec. 5.3 for discussion on the cost of constructing the reduced matrix). The number of floating point operations per iteration is approximately twice the number of nonzeros in each matrix. The number of nonzero entries in the reduced matrix is higher by a factor of about 35\% than the number of nonzeros for the unreduced matrix. For the block Jacobi method, in the LU decomposition of the preconditioner (which is done once and for all), there is no fill-in for the unreduced matrix, and there is fill-in of a modest amount of \(\frac{a^3}{2}\) entries for the reduced matrix.
Chapter 4. Convergence Analysis for Block Stationary Methods

As a result, each iteration for the reduced system costs approximately $20n^3$ floating point operations, and each iteration for the unreduced matrix costs approximately $14n^3$ floating point operations. The iterations in the reduced solver are thus approximately 42% more expensive.

<table>
<thead>
<tr>
<th>system</th>
<th>reduced</th>
<th>unreduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDE coeff. ($\sigma = \tau = \mu$)</td>
<td>10 20 100 1000</td>
<td>10 20 100 1000</td>
</tr>
<tr>
<td>Jacobi centered</td>
<td>393 173 53 N/C</td>
<td>1,030 444 N/C N/C</td>
</tr>
<tr>
<td>GS centered</td>
<td>188 77 14 322</td>
<td>492 198 N/C N/C</td>
</tr>
<tr>
<td>SOR centered</td>
<td>36 25 - -</td>
<td>61 38 - -</td>
</tr>
<tr>
<td>Jacobi upwind</td>
<td>455 239 75 43</td>
<td>1,194 620 179 89</td>
</tr>
<tr>
<td>GS upwind</td>
<td>219 111 27 10</td>
<td>574 287 63 16</td>
</tr>
<tr>
<td>SOR upwind</td>
<td>39 27 18 9</td>
<td>66 45 24 11</td>
</tr>
</tbody>
</table>

Table 4.4: comparison between iteration counts for the reduced and unreduced systems, for different values of mesh Reynolds numbers. 'N/C' marks no convergence after 2,000 iterations. The experiments were performed on a $32 \times 32 \times 32$ grid.

For the values of the PDE coefficients in the table, namely 10, 20, 100 and 1000, the corresponding values of the mesh Reynolds numbers are 0.1515, 0.3030, 1.515 and 15.15. Notice that the last two are larger than 1, and so, for these values we do not know the optimal relaxation parameter and the SOR experiments for these values were not performed.

The following observations can be made:

1. Overall the reduced solver is substantially faster than the unreduced solver. Even though each iteration of the reduced solver is more expensive (as explained above), the savings in iteration counts are much more significant and in regions where both solvers converge, the reduced solver is more efficient by a factor of at least 50% (except one case), and much more in many cases. Moreover, a significant fact here is that there are cases where the reduced solver converges whereas the unreduced solver does not.

2. For the reduced solver, the Gauss-Seidel scheme outperforms the Jacobi scheme by a factor of approximately 2 in diffusion-dominated regions, which illustrates the “near Property
Chapter 4. Convergence Analysis for Block Stationary Methods

A matrix has. In convection-dominated regions with centered differencing the Gauss-Seidel scheme is significantly better than Jacobi. (In this case the matrix is not necessarily nearly consistently ordered).

3. If centered differences are used, for \( \sigma = 20 \) convergence is faster than for \( \sigma = 10 \). This illustrates the phenomenon which is supported by the analysis and holds also for the two-dimensional case [41] - that for sufficiently small mesh Reynolds numbers, the “more nonsymmetric” systems converge faster than the “close to symmetric” ones.

4. The upwind difference scheme converges more slowly than the centered difference scheme when the mesh Reynolds numbers are small in magnitude, but convergence is fast for large mesh Reynolds numbers. This applies to both the reduced and the unreduced systems, and follows from the fact that as the PDE coefficients grow larger, the matrix is more diagonally dominant when upwind schemes are used.

Next, we consider an example of a nonseparable problem, for which our convergence analysis does not apply. Consider the problem

\[-0.1 \Delta u + yz u_x + xz u_y + xy u_z = w,\]

with Neumann boundary conditions \( u_z = 0 \) at \( z = 0, 1 \), and zero Dirichlet conditions for \( x = 0, 1 \) and \( y = 0, 1 \), on the unit cube. Note that there is a turning point, but it is on the boundary.

Here \( w \) was constructed so that the exact solution is \( u(x, y, z) = \sin(\pi x) \sin(\pi y) \cos(\pi z) \). The ordering strategy used is 2PNxz. In this case it is impossible to know the optimal SOR parameter; we examine the iteration count for block Jacobi and block Gauss-Seidel. The results in Table 4.5 are for a \( 20 \times 20 \times 20 \) grid (8,000 gridpoints in the tensor-product grid), using centered difference discretization.

The following observations can be made:

1. Even though the problem is nonseparable, the solvers have a behavior which is similar to the convergence results for the constant coefficient case. Refer back to Eq. (4.68),
Table 4.5: iteration counts for different iterative schemes

(4.69), (4.94), (4.96): The difference in iteration counts matches the ratios predicted by the analysis; the improvement after the cyclic reduction step is performed is by a factor of 100% to 200% in iteration counts. As is predicted by the analysis, the improvement for the 1D solver is more dramatic than the improvement for the 2D solver.

2. For the unreduced system the differences between the iteration counts between the 1D solvers and the 2D solvers is approximately a factor of 1.5; in the constant coefficient case the convergence analysis provides a factor of 1.8. Also, we notice an improvement by a factor of approximately 2 between the 1D and the 2D solver (in iterations) for the unreduced system; the predicted factor of improvement for the constant coefficient case is 2.

3. The improvement in rate of convergence between Jacobi and Gauss-Seidel is approximately 2 for 1D partitioning, relative to which the matrix does not have Property A.

Comparison of variants of the orderings

Tables 4.6 and 4.7 present selected results of applying block Gauss-Seidel based on various splittings. These results were obtained as part of an extensive set of numerical experiments which have been performed for all possible combinations of signs of the PDE coefficients, for various magnitudes of the convective terms and various test problems. The results presented in the tables are for the model problem (1.15), where the right-hand-side vector is such that the analytical solution is $u = 100xyz(1 - x)(1 - y)(1 - z) \exp(x + y + z)$. Centered difference discretization was done on a $12 \times 12 \times 12$ grid. The results give good indication as to the situation
Chapter 4. Convergence Analysis for Block Stationary Methods

for finer grids. The stopping criterion was \( \|r^{(i)}\|_2/\|r^{(0)}\|_2 \leq 10^{-13} \). The two-plane matrix was used, with \( x-z \) oriented sets of gridpoints. The following partitionings were considered:

1. The natural two-plane ordering with 1D preconditioning (termed '1D' in the tables).

2. The four-color block ordering with 1D preconditioning ('4C').

3. The natural two-plane ordering with 2D preconditioning ('2D').

4. The 2D red/black block ordering with 2D preconditioning ('RB').

<table>
<thead>
<tr>
<th>( \sigma )</th>
<th>( \tau )</th>
<th>( \mu )</th>
<th>1D</th>
<th>4C</th>
<th>2D</th>
<th>RB</th>
<th>( \sigma )</th>
<th>( \tau )</th>
<th>( \mu )</th>
<th>1D</th>
<th>4C</th>
<th>2D</th>
<th>RB</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>15</td>
<td>12</td>
<td>11</td>
<td>500</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>14</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>-100</td>
<td>0</td>
<td>0</td>
<td>17</td>
<td>15</td>
<td>12</td>
<td>11</td>
<td>-500</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>14</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>100</td>
<td>0</td>
<td>24</td>
<td>26</td>
<td>28</td>
<td>28</td>
<td>0</td>
<td>500</td>
<td>0</td>
<td>197</td>
<td>201</td>
<td>192</td>
<td>194</td>
</tr>
<tr>
<td>0</td>
<td>-100</td>
<td>0</td>
<td>28</td>
<td>25</td>
<td>30</td>
<td>28</td>
<td>0</td>
<td>-500</td>
<td>0</td>
<td>203</td>
<td>202</td>
<td>194</td>
<td>195</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>100</td>
<td>25</td>
<td>24</td>
<td>12</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>500</td>
<td>197</td>
<td>201</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>-100</td>
<td>28</td>
<td>24</td>
<td>12</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>-500</td>
<td>202</td>
<td>201</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 4.6: iteration counts for the block Gauss-Seidel scheme, for one nonzero convective term.

<table>
<thead>
<tr>
<th>( \sigma )</th>
<th>( \tau )</th>
<th>( \mu )</th>
<th>1D</th>
<th>4C</th>
<th>2D</th>
<th>RB</th>
<th>( \sigma )</th>
<th>( \tau )</th>
<th>( \mu )</th>
<th>1D</th>
<th>4C</th>
<th>2D</th>
<th>RB</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>100</td>
<td>0</td>
<td>28</td>
<td>26</td>
<td>26</td>
<td>26</td>
<td>500</td>
<td>500</td>
<td>0</td>
<td>260</td>
<td>273</td>
<td>254</td>
<td>275</td>
</tr>
<tr>
<td>-100</td>
<td>100</td>
<td>0</td>
<td>28</td>
<td>26</td>
<td>25</td>
<td>26</td>
<td>-500</td>
<td>500</td>
<td>0</td>
<td>253</td>
<td>278</td>
<td>256</td>
<td>279</td>
</tr>
<tr>
<td>-100</td>
<td>-100</td>
<td>0</td>
<td>31</td>
<td>26</td>
<td>28</td>
<td>26</td>
<td>-500</td>
<td>-500</td>
<td>0</td>
<td>263</td>
<td>278</td>
<td>263</td>
<td>280</td>
</tr>
<tr>
<td>100</td>
<td>-100</td>
<td>0</td>
<td>31</td>
<td>26</td>
<td>27</td>
<td>26</td>
<td>500</td>
<td>-500</td>
<td>0</td>
<td>264</td>
<td>274</td>
<td>258</td>
<td>275</td>
</tr>
</tbody>
</table>

Table 4.7: iteration counts for the block Gauss-Seidel scheme, for two nonzero convective terms.

The results can be interpreted in two aspects: performance of natural block orderings vs. multicolor block orderings, and relation to the direction of the velocity vectors. In [39], Elman & Chernesky have studied the one-dimensional convection-diffusion equation in conjunction with the Gauss-Seidel solver, and concluded that the solver is most effective when the sweeps
follow the flow; when the sweeps are opposite to the flow, red/black orderings perform better than natural orderings, but not as well as natural orderings in the case where the sweeps follow the flow. In [42], Elman & Golub conducted numerical experiments for the two-dimensional convection-diffusion problem, using one-line and two-line orderings in conjunction with block stationary methods as well as ILU preconditioned GMRES, and for stationary methods concluded that red/black orderings have essentially the same iteration counts in average as natural orderings have, but they are less sensitive to the direction of the flow.

For the three-dimensional problem with the orderings examined here, we observe that in general multicolor and natural orderings have essentially the same iteration counts for moderate convection (PDE coefficients of magnitude 100) but on the other hand, the performance of natural orderings is better when convection dominates. (The differences are bigger when all convective terms are nonzero.) Multicolor orderings are less sensitive to change of sign of the PDE coefficients. Loosely speaking, if good performance is related to following the direction of the flow, then since the 1D blocks are $x$-$z$ oriented and the ordering of the blocks in $y$ direction is done from bottom to top, it is expected that when the signs of $\tau$ or $\mu$ are positive, the scheme would converge faster (the sign of $\sigma$ plays a smaller role, as it is associated with the variable in the direction of which the preconditioning is done). As can be observed in Table 4.6, the natural orderings are slightly more sensitive to the change of sign in $\tau$ (orientation of 2D sets of gridpoints), and least sensitive to changes in $\sigma$ (the direction associated with the preconditioning). Since there is alternating direction in the ordering of the gridpoints in each of the 1D sets of gridpoints, the sensitivity to sign is in general not particularly high.

The fact that the Gauss-Seidel solver with multicolor orderings converges in a reasonable rate is meaningful; these ordering strategies are parallelizable (see Sec. 5.5).
Chapter 5

Solvers, Preconditioners, Implementation and Performance

In this chapter solving techniques, implementation and performance are addressed. For solving the reduced system, block stationary methods have been considered in Chap. 4, and in this chapter we consider preconditioned Krylov subspace solvers. This is followed by discussion on implementation. We then present numerical results which demonstrate the effectiveness of solvers of the cyclically reduced system.

5.1 Krylov Subspace Solvers and Preconditioners - Overview

Krylov subspace solvers are a family of projection methods, which in the context of linear systems can be described as methods that compute an approximation to the solution \( x = A^{-1}b \) in a particular subspace of \( \mathbb{R}^n \). Requiring that the approximate solution belong to a specific subspace introduces a series of constraints. These could be, for example, orthogonality conditions on the residual. A general introduction to projection methods can be found in [95, Chap. 5].

Krylov subspace methods have not been used as long as stationary methods. Golub and van der Vorst [58] remark that the first Krylov subspace methods were developed roughly at the same time the SOR method was developed, but the latter was more popular, due to its modest storage requirements. The available convergence analysis of these solvers is less comprehensive than the convergence analysis of stationary methods, particularly for nonsymmetric systems like ours. Nevertheless, their performance is generally very good, particularly when the linear
system is effectively preconditioned first. Much research work has been done in this area in the last ten to fifteen years; many new variants and convergence results have been derived, and numerical difficulties (in particular, situations of breakdown) have been addressed. There are several excellent references for these methods. Among them we mention the books of Saad [95] and Greenbaum [61], and the survey papers of Freund, Golub & Nachtigal [49] and Golub & van der Vorst [58].

As before, consider the linear system $Ax = b$, and let $r^{(k)} = b - Ax^{(k)}$ be the residual at the $k$th iterate. The $k$-dimensional Krylov subspace associated with $A$ and $r^{(0)}$ is defined as:

$$
K_k(A, r^{(0)}) = \text{span}\{r^{(0)}, Ar^{(0)}, \ldots, A^{k-1}r^{(0)}\}.
$$

The basic principle is: at the $k$th iterate, seek an approximate solution $x^{(k)}$ in the shifted Krylov subspace $x^{(0)} + K_k(A, r^{(0)})$. The criteria for computing $x^{(k)}$ fall into three main categories [58]:

1. **The Ritz-Galerkin approach**: compute $x^{(k)}$ such that $r^{(k)}$ is orthogonal to $K_k(A, r^{(0)})$. For general nonsymmetric systems this is known as FOM (full orthogonalization method) [96]. When the matrix is symmetric positive definite we get the well known conjugate gradient method [69].

2. **The minimum residual approach**: compute $x^{(k)}$ such that $\|r^{(k)}\|_2$ is minimal over $K_k(A, r^{(0)})$.

In this category MINRES [85] and GMRES [96] are well known methods.

3. **The Petrov-Galerkin approach**: compute $x^{(k)}$ so that $r^{(k)}$ is orthogonal to some other $k$-dimensional subspace. BiCG [90] and QMR [50] are methods based on this approach.

Highly effective methods that are hybrids of the above three approaches are CGS [100], Bi-CGSTAB [110], TFQMR [48], FGMRES [93], and others - see [58] for descriptions.

The first concern when computing optimal solutions in the Krylov subspace is the basis that should be used. Using the obvious basis $\{A^j r^{(0)}\}$ would cause numerical difficulties in finite precision arithmetic, as the vectors $A^j r^{(0)}$, even for relatively small $j$, might be dominated (as far as their direction goes) by the eigenvector corresponding to the largest eigenvalue; instead, an orthonormal basis is formed. A widely used technique for generating this basis is Arnoldi’s
method [5], which was originally introduced as a technique for reducing a dense matrix into an upper Hessenberg form (useful for computing the eigenvalues). At each step, the $j$th vector $v_j$ of the basis for $K_k$ is multiplied by the matrix $A$, and orthogonalized against all previous vectors in the basis, namely $v_1, \ldots, v_{j-1}$. The starting vector is

$$v_1 = \frac{r^{(0)}}{\|r^{(0)}\|_2}.$$  \hfill (5.2)

The upper Hessenberg matrix is generated throughout the process.

Originally, the technique for orthogonalizing the vectors in [5] was the standard Gram-Schmidt algorithm. Later versions of Arnoldi’s method use modified Gram-Schmidt, which is mathematically equivalent to Gram-Schmidt but is more stable numerically, and Householder transformations (see [55] for details on computational work and accuracy of these methods).

When the matrix $A$ is nonsymmetric, the process of orthogonalization becomes increasingly expensive as the dimension of the subspace increases. Denoting the $(k+1) \times k$ upper Hessenberg matrix generated in Arnoldi’s algorithm by $H_{k+1,k}$, we have [95]:

$$V_k^T A V_k = H_{k,k},$$ \hfill (5.3)

where $V_k$ is the $k \times k$ matrix whose columns are the vectors of the orthonormal basis, and $H_{k,k}$ is the matrix obtained by taking the first $k$ rows of $H_{k+1,k}$. From here, it depends which approach is taken. The Ritz-Galerkin approach is equivalent to requiring $V_k^T r^{(k)} = 0$, and it can be shown [95] that $x^{(k)} = x^{(0)} + V_k y^{(k)}$, where $y^{(k)} = [H_{k,k}]^{-1} \|r^{(0)}\|_2^2 e_1$, and $e_1$ denotes the first vector in the standard basis for $\mathbb{R}^k$. If $A$ is symmetric positive definite, the Hessenberg matrix is reduced to a tridiagonal form, and the above-described procedure is (essentially) the conjugate gradient method [69] (see also [56]).

If the above approach is modified so that instead of orthogonality, one requires that $\|r^{(k)}\|_2$ is minimal over the shifted Krylov space, then the result is the GMRES method of Saad & Schultz [96]. This is a most important contribution, which actually marked the beginning of a wave of enormous popularity of Krylov subspace methods. Practical implementation of the algorithm is discussed in detail by Saad in [95, Sec. 6.5.3]. For a theoretical discussion on the
The Petrov-Galerkin approach is based on the attempt to obtain the attractive property of three-term recurrence relations among the eigenvectors (that symmetric matrices have) “by brute force”. In general, an orthogonal basis based on three-term recurrence relations cannot be constructed for nonsymmetric matrices. Nevertheless, it can be obtained if the requirement of orthogonality is dropped. One can then add constraints of orthogonality with respect to some other basis. This means that we are looking for a matrix $W_i$, such that each new basis vector $v_i$ of the Krylov subspace is orthogonal to the first $i - 1$ column vectors of $W_i$ ($w_1, \ldots, w_{i-1}$), and we also require $v_i^T w_i \neq 0$. The construction is thus done by generating bi-orthogonal basis sets. The BiCG [90] and the QMR [50] methods are based on these ideas (with features that are designed to avoid certain situations of breakdown). For example, in the BiCG method there are two sequences of residuals, one for each of the (bi-orthogonal) sets, denoted by $r^{(j)}$ and $\tilde{r}^{(j)}$, which are polynomials in $A$ and $A^T$ respectively.

As mentioned earlier, hybrids between the various approaches have been also developed. One of the most popular methods here is Bi-CGSTAB, of van der Vorst [110]: it is a variant of the BiCG method, derived by a different choice of the polynomial associated with $\tilde{r}^{(j)}$: the choice of the polynomial coefficients is done so that the residual vector $r^{(j)}$ is minimized. In this sense Bi-CGSTAB is a hybrid of BiCG and GMRES.

Krylov subspace methods perform significantly better when the matrix is well conditioned. An illustration of that can be given by referring to the following convergence result for the conjugate gradient method [61]:

$$
\| x - x^{(k)} \|_A \leq 2 \cdot \left( \frac{\sqrt{\kappa_2(A)} - 1}{\sqrt{\kappa_2(A)} + 1} \right)^k \cdot \| x - x^{(0)} \|_A ,
$$

where $\| \cdot \|_A$ is the energy norm, defined by $\| y \|_A = \sqrt{y^T A y}$. As is evident, convergence is fast when $A$ is well conditioned; when it is not, it is worthwhile to find a preconditioning matrix, say $M$, which has the property that $M^{-1} A$ is better conditioned than $A$, and then replace the
original system $Ax = b$ by

$$M^{-1}Ax = M^{-1}b . \quad (5.5)$$

Effective preconditioning is a crucial issue. Saad [95, p. 245] states that “In general, the reliability of iterative techniques, when dealing with various applications, depends much more on the quality of the preconditioner than on the particular Krylov subspace accelerators used”.

In fact, we have already discussed preconditioned systems in Chap. 4; indeed, for fixed point schemes based on the splitting $A = M - N$ we have $x^{(k+1)} = g(x^{(k)})$, where $g : \mathbb{R}^n \to \mathbb{R}^n$, and

$$g(x) = M^{-1}Nx + M^{-1}b . \quad (5.6)$$

Convergence is to the fixed point of $g$:

$$x = g(x) , \quad (5.7)$$

and thus by substituting (5.7) and $N = M - A$ in (5.6) we are actually solving the linear system (5.5).

Two obvious ways to perform preconditioning are [95]:

1. *left* preconditioning, which means solving (5.5).

2. *right* preconditioning, which means solving $AM^{-1}y = b$, $y = Mx$.

The difference between these two approaches is discussed in [95, Sec. 9.3.4]. Combination of left and right preconditioning is also widely used (split preconditioning). Among popular preconditioning techniques we mention preconditionings based on incomplete factorizations, preconditionings based on approximate inverses, and block preconditioners. See [95] for an overview of several of these techniques.

One of the most popular techniques for preconditioning a linear system is the class of *incomplete factorizations*. These methods are based on computing a factorization $LU$ which is a fairly good approximation to the matrix, and at the same time the factors are sparse, so that the preconditioner solve is inexpensive. Here the simplest forms are (point or block) Jacobi
and SSOR. These factorizations are based on the coefficient matrix itself: Jacobi corresponds to diagonal preconditioning, and SSOR corresponds to taking \( L = I + \omega ED^{-1} \) and \( U = D + \omega F \), where \( D \), \( E \) and \( F \) are the diagonal, lower triangular and upper triangular parts of the matrix respectively. If we set \( \omega = 1 \) we get the SGS scheme. These preconditioning techniques are set-up in minimal cost (SSOR) or no cost at all (Jacobi). No additional storage is required for both techniques. On the other hand, these techniques (and in particular the Jacobi preconditioner) are not always effective (see numerical examples in [87]).

Incomplete LU (ILU) factorizations, which are based on computing the LU decomposition, but without allowing much fill-in to occur, form a popular class of methods. In 1977, Meijerink and van der Vorst [78] showed that when the matrix for which the factorization is performed is a symmetric \( M \)-matrix, there exists an ILU factorization and it is highly effective when applied to conjugate gradient as a preconditioner. This paper was very important in understanding the merits of incomplete LU factorizations and in directing interest of researchers to these techniques.

A well known member in the class of incomplete LU factorizations is \( \text{ILU}(0) \); here \( L \) and \( U \) are constructed so that the matrix \( \bar{A} = LU \) satisfies \( \bar{a}_{i,j} = a_{i,j} \) whenever \( a_{i,j} \neq 0 \). \( \text{ILU}(0) \) is easy to implement. Generally, the cost of setting it up is approximately

\[
C = \frac{\bar{m}^2}{\bar{n}} \tag{5.8}
\]

floating point operations [87], where \( \bar{m} \) is the number of nonzeros in the matrix and \( \bar{n} \) is the number of unknowns.

\( \text{ILU}(0) \) could behave poorly, for several reasons [14]. One possible reason: the pivots can be very small. In nonsymmetric matrices arising from discretization of PDEs, diagonal dominance is not at all guaranteed. In fact, even when the coefficient matrix is very well behaved, for example symmetric positive definite, the pivots of the incomplete factorization are not guaranteed to be necessarily positive [11], or far from zero in magnitude. Another source of trouble could be that the triangular solvers themselves are severely ill-conditioned. Elman [38] studied this aspect in detail. Ways of detecting the ill-conditioning are suggested in [24].
Chapter 5. Solvers, Preconditioners, Implementation and Performance

The conditioning of the factors for certain two-dimensional convection-diffusion equations has been experimentally studied in [14].

One way to improve the stability is by allowing more nonzeros in the factors $L$ and $U$. For example, denote the factors of the ILU(0) factorization by $L_0$ and $U_0$. Then, the ILU(1) factorization is obtained by allowing $L$ and $U$ have the nonzero structure of $L_0 U_0$. The ILU(1) factorization is typically more accurate compared to ILU(0). On the other hand, the number of nonzeros can be substantially higher, so that the process is more costly (a system associated with the preconditioning matrix has to be solved once or twice at every iteration). The same principle leads to ILU(2), ILU(3), and so on.

The number of nonzeros in the factors is important, because it can serve as a good indication for:

1. The storage required.

2. The set-up time for the preconditioner.

3. The time a preconditioner solve requires.

Obviously, the number of nonzeros in the factors of the ILU(0) factorization is equal to the number of nonzeros in the sum of the triangular parts of the original matrix. In order to find the nonzeros of the matrices associated with the more accurate ILU(1) factorization, there is no need to actually perform any matrix product [95]. Instead, one can determine the fill-in by examining the computational molecules associated with the triangular parts $L$ and $U$. ILU(p) factorizations exist if the matrix is an $M$-matrix [78], and in this case their associated splitting is a regular splitting [95, Thm. 10.2].

ILU(p) factorizations do not take into account the magnitude of the matrix elements. One might instead consider using a threshold: look at the magnitude of each element during the factorization, and drop it if it is below a certain (prespecified) threshold. Such a technique is ILUT [94], which first applies a rule based on threshold, and after the factorization is completed takes only a certain number of elements (the largest) in each row. When the drop tolerance
is sufficiently small these factorizations are very effective (see, for example, Pommerell [87]); however, the amount of nonzeros in the factors could be substantially higher than that of the coefficient matrix.

All the above-mentioned incomplete factorizations have block versions. The idea of these factorizations is to treat the matrix as consisting of blocks (submatrices) rather than elements. The existence of incomplete block factorizations is guaranteed if the matrix is an $M$-matrix (Axelsson [8]). An important paper which contains a comprehensive testing of various block preconditioners for the symmetric positive definite case is by Concus, Golub & Meurant [26]. When using block factorizations, inverting the pivotal blocks is costly and might cause fill-in, as the inverse of a sparse matrix is not necessarily sparse (see Meurant [79] for a review of the topic of inverses of symmetric tridiagonal and block tridiagonal matrices). There is fill-in also in the off-diagonal blocks, due to a repetitive computation of Schur complements of the blocks. In order to overcome the problem of extra work and storage, approximate inverses which preserve sparsity are considered. Possible strategies are suggested, for example, in [9], [34], [114]. If the matrix is referred to as block tridiagonal then only the diagonal block elements are modified and fill-in can be avoided. Modifying only the diagonal elements can be done also for general matrices (not necessarily tridiagonal), and is known as the D-ILU factorization (Pommerell [87]).

In general, block incomplete LU factorizations are less effective for three-dimensional problems than for two-dimensional problems [11]. Magolu & Polman [76] discuss the effectiveness of block ILU factorizations for three-dimensional problems, and show that line partitionings could in some cases be superior to point incomplete factorizations, but not always; plane partitionings are less effective.

5.2 Incomplete Factorizations for the Reduced System

For our reduced system, recall that the computational molecule consists of 19 points. The $i$th row of $LU$ is obtained by multiplying each of the ten components of the computational molecule
which are associated with the matrix $L$, by the corresponding rows in $U$. Suppose the row of the matrix for which fill-in is examined is associated with a gridpoint whose coordinates are $(i, j, k)$, and numbering of the unknowns is done in $x$-$y$ plane oriented natural lexicographic fashion. In Fig. 5.1 we present a two-dimensional slice of the stencil associated with the factor $U$; we pick the plane where the associated gridpoint for which discretization was done lies. The stencil in the figure is also the fill-in pattern corresponding to the planes which contain gridpoints $(i, j, k \pm 2)$.

\[
\begin{array}{ccc}
X \\
X & X \\
\_X & X \\
\end{array}
\]

Figure 5.1: a two-dimensional slice of the stencil associated with $U$ (the value on the diagonal is circled)

In Fig. 5.2 we demonstrate the fill-in (a 2D slice) that occurs by combining the stencils in Fig. 5.1 associated with all the components in the computational molecule. In Fig. 5.3 the stencils associated with the neighboring planes are presented.

\[
\begin{array}{ccc}
X & X \\
X & X & X \\
\_X & \_X & X \\
\_X & \_X & X \\
\_X & X \\
\end{array}
\]

Figure 5.2: fill-in in the construction of ILU(1) in the plane that contains the gridpoint for which the discretization was done (dashed circle). Circled are the points that belong to the original computational molecule.

Having determined the nonzero patterns in all planes which belong to the lower triangular
matrix $L$, where fill-in is experienced, the above can be summarized as follows:

**Proposition 5.1.** Denote by $L_1$ and $U_1$ the matrices associated with the ILU(1) factorization of the reduced matrix associated with lexicographic ordering. Then the number of nonzeros of $L_1 + U_1$ satisfies

$$\text{nz}(L_1 + U_1) \approx \frac{45}{19} \cdot \text{nz}(S),$$

where $\text{nz}(S)$ stands for the number of nonzeros in the reduced matrix.

*Proof.* Fig. 5.1-5.3 clarify how many nonzeros are generated in each 2D slice when going by planes $z$-direction. (There are other 2D slices that need to be considered, but these do not generate any nonzeros that have not been already presented in the three figures; the details are omitted.) Thus in each row associated with an interior gridpoint there are $13 + 11 \cdot 2 + 5 \cdot 2 = 45$ nonzeros. The two factors of 2 here come from the fact that there are two neighboring planes, and two separate planes contain the points $(i, j, k \pm 2)$. For points next to the boundary the fill-in level is smaller, and thus the the number $\frac{45}{19}$ is only an estimate.

In order to check the estimate in Prop. 5.1, the ILU(1) factorizations of reduced matrices corresponding to $8 \times 8 \times 8$, $16 \times 16 \times 16$ and $24 \times 24 \times 24$ grids have been generated; the actual number of nonzeros and the estimates are presented in Table 5.1.
Chapter 5. Solvers, Preconditioners, Implementation and Performance

<table>
<thead>
<tr>
<th>n</th>
<th>nz(S)</th>
<th>nz($L_1 + U_1$)</th>
<th>estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$8 \times 8 \times 8$</td>
<td>3,760</td>
<td>7,522</td>
<td>8,905</td>
</tr>
<tr>
<td>$16 \times 16 \times 16$</td>
<td>34,400</td>
<td>75,218</td>
<td>81,474</td>
</tr>
<tr>
<td>$24 \times 24 \times 24$</td>
<td>121,104</td>
<td>272,194</td>
<td>286,825</td>
</tr>
</tbody>
</table>

Table 5.1: number of nonzero elements in the ILU(1) factorization: actual numbers and estimates

The same can be done for any other ordering, and for higher allowed levels of fill. In Fig. 5.4 the sparsity patterns of the factors corresponding to the ILU(1) and the ILU(2) factorizations for the two-plane matrix are presented.

Figures 5.4: sparsity patterns of the factors of ILU(1) and ILU(2), associated with the two-plane matrix.

Multicolor block versions of orderings, discussed in Chap. 3, give rise to a larger amount of fill-in in the incomplete factorization. For the four-color two-plane ordering the matrix has 8,700 nonzero - compare to 7,510 for the natural two-plane ordering. As a result, more work per iteration is required.

The following result is due to Beaudens [13]:

123
Suppose $A$ is a nonsingular $M$-matrix and $A = D_1 - C_1 = D_2 - C_2$ are splittings of $A$, and let $D_1 = L_1 U_1$ and $D_2 = L_2 U_2$ be incomplete factorizations of $A$ such that the set of nonzeros of $L_1 + U_1$ is contained in the set of nonzeros of $L_2 + U_2$. Then

\[ \rho(D_2^{-1}C_2) \leq \rho(D_1^{-1}C_1) . \]  

(5.10)

This result was used by Elman & Golub in [42, Thm. 2], and can also be directly used for our reduced system:

**Proposition 5.2.** If the reduced matrix is an $M$-matrix, the spectral radius of the iteration matrix associated with $ILU(p)$ factorizations is smaller than or equal to the spectral radius of the iteration matrix associated with block Jacobi with the 2D splitting.

Prop. 5.2 applies, for example, to the constant coefficient case where $be, \ cd, \ fg > 0$. For the model problem we can also obtain an existence result for the ILUT factorization. For showing this, we need the following [95, p. 290]:

**Definition 5.1.** An $N \times N$ matrix $H$ is an $\hat{M}$-matrix if its last entry on the diagonal is non-negative, the rest of the diagonal values are all positive, the off-diagonal values are nonpositive, and

\[ \sum_{j=i+1}^{N} h_{i,j} < 0 , \quad 1 \leq i < N . \]  

(5.11)

**Proposition 5.3.** Consider the model problem and suppose that either upwind differences are used, or centered differences with $be, \ cd, \ fg > 0$. Then the ILUT factorization exists for the reduced matrix (for all the orderings that have been considered in Chap. 3).

**Proof.** The diagonal values of the matrix are $a^2 - 2be - 2cd - 2fg$ in rows associated with interior points, and larger in rows associated with with non-interior points. Since the matrix is an $M$-matrix in the cases specified in the lemma, all its off-diagonal elements are nonpositive. The matrix has at least one negative value to the right of the main diagonal in each row except

124
Chapter 5. Solvers, Preconditioners, Implementation and Performance

the last. Thus Eq. (5.11) holds and the matrix is an M-matrix. In addition, since the matrix is
diagonally dominant, and moreover, since the diagonal entries are all positive, it is guaranteed
that all its row sums are nonnegative. This special property is termed in [95] a diagonally
dominant M-matrix. The matrix thus satisfies all the conditions required for existence of the
ILUT factorization, by [95, Thm. 10.13].

In Fig. 5.5 the nonzero pattern of the factors corresponding to ILU factorization with drop
tolerance $10^{-2}$ is presented. Here the factorization was obtained by using Matlab’s command
‘luinc’, where the dropping rule is: for each entry, the threshold is the drop tolerance specified
by the user, multiplied by the norm of the column containing the entry (with the exception
that diagonal values of the matrix $U$ are never dropped). The matrix which is shown in the
figure comes from centered difference discretization of the equation $-\Delta u + 10(x, y, z)^T \nabla u = f$.

Note that for ILU factorizations based on thresholds, there is flexibility (which does not exist
in ILU(p) factorizations) which allows more fill-in for more ill-conditioned problems.

![Figure 5.5: sparsity patterns of factors for ILU with drop tolerance $10^{-2}$.](image)

Block factorizations based on 1D partitioning require the following amount of computational
work: the number of block rows is $\frac{n^2}{4}$, and the semibandwidth of the matrix is $n + 1$. The
number of nonzero blocks in the lower part as well as in the upper part of this matrix is
$n^2 + O(n)$. Typically there are 9 non-zero block elements in each row. Each element is a
Chapter 5. Solvers, Preconditioners, Implementation and Performance

A $2n \times 2n$ matrix. In total there are approximately $5n^2$ operations involving $2n \times 2n$ matrix products and $n^2$ inversions or approximate inversions of $2n \times 2n$ matrices. Taking the exact inverses of the diagonal blocks, and considering the fill-in in off-diagonal block elements, the total cost is $O(n^5)$ operations, which is not satisfactory. Variants which restrict the fill-in in the off-diagonals blocks are possible but have not been tested. Some tests with incomplete LU factorizations based on 2D partitioning with approximate inverses of the pivotal blocks have been performed and have not been found particularly effective.

5.3 The Overall Cost of Construction of the Reduced System

One important issue in the implementation of one step of cyclic reduction is the cost of performing it. It will now be shown that the construction of the reduced system is an inexpensive component compared to the work involved in computing the solution. Nevertheless, it is not negligible.

Consider first the constant coefficient case. Here the reduced matrix can be constructed in a straightforward manner, and there is no need to actually construct the unreduced matrix first and perform the Gaussian elimination. By the difference equation (2.7) it is evident that since the computational molecule is gridpoint-independent there are merely 2-3 floating point operations for each of the 19 components of the computational molecule (and fewer floating point operations in total for points next to the boundary). The overall computational work is thus negligible; in fact most of the construction time is spent on assigning the values of the computational molecule to the matrix. Since the reduced matrix has a block structure similar to the block structure of the unreduced matrix, the overall construction work is almost equal to the work involved in constructing the latter.

The variable coefficient case requires more careful attention. Here each entry in the matrix is computed separately, as the values of the computational molecule are gridpoint-dependent. For both the reduced and the unreduced systems the set-up of the components of the computational molecule throughout the grid needs to be done. First, the PDE coefficients for every gridpoint
need to be computed. There are six trivariate functions, each of which needs to be evaluated at \( n^3 \) gridpoints (or approximately this number). The cost of this step depends on the cost of evaluating the PDE coefficients and could be high. In addition, there is a need to construct the components of the computational molecule. By Eq. (2.13) and (2.14), this takes approximately \( 125n^3 \) operations for centered differences, and \( 105n^3 \) operations for upwind differences. As for the step of cyclic reduction, counting operations in Eq. (2.15), there is a total of about 125 floating point operations for computing the entries of the matrix in a row that corresponds to an interior gridpoint. However, certain values in the difference equation appear several times; each of the values \( \frac{b_{i,j,k}}{a_{i,j-1,k}}, \frac{c_{i,j,k}}{a_{i,j-1,k}}, \frac{d_{i,j,k}}{a_{i,j+1,k}}, \frac{e_{i,j,k}}{a_{i,j+1,k}}, \frac{g_{i,j,k}}{a_{i,j,k+1}} \) appears six times in the difference equation, and careful implementation reduces the number of floating point operations by 20%-25%. Even if the construction is done without such optimization (which is the case if block Gaussian elimination is applied directly), since the matrix is of size \( \frac{n^3}{2} \times \frac{n^3}{2} \), the overall amount of floating point operations for constructing the reduced matrix is approximately \( 60n^3 \). The construction time of the right-hand-side vector is approximately \( 3n^3 \). In Table 5.2 we present the actual amount of work that was required for the overhead of constructing the reduced matrix for a few grids, using Gaussian elimination.

<table>
<thead>
<tr>
<th>( n )</th>
<th>( n^3 )</th>
<th>Mflops</th>
<th>( \text{flops} ) ( n^3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>512</td>
<td>0.025</td>
<td>48.8</td>
</tr>
<tr>
<td>12</td>
<td>1,728</td>
<td>0.090</td>
<td>52.1</td>
</tr>
<tr>
<td>16</td>
<td>4,096</td>
<td>0.222</td>
<td>54.2</td>
</tr>
<tr>
<td>20</td>
<td>8,000</td>
<td>0.442</td>
<td>55.3</td>
</tr>
<tr>
<td>24</td>
<td>13,824</td>
<td>0.775</td>
<td>56.1</td>
</tr>
<tr>
<td>28</td>
<td>21,952</td>
<td>1.243</td>
<td>56.6</td>
</tr>
<tr>
<td>32</td>
<td>32,768</td>
<td>1.869</td>
<td>57.0</td>
</tr>
</tbody>
</table>

Table 5.2: computational work involved in the construction of the reduced matrix.

The reduced system is constructed once and for all, and the construction is clearly an inexpensive component compared to setting up a preconditioner (approximately \( 200n^3 \) floating point operations for setting up the \( \text{ILU}(0) \) factorization) or solving the system (at least \( 80n^3 \)
operations \textit{per iteration} for Krylov subspace solvers like Bi-CGSTAB or CGS). Finally, it is noted that the construction can be almost fully parallelized or vectorized, as illustrated in Sec. 5.5.

### 5.4 Numerical Experiments

This section provides details on numerical experiments. Results of some numerical experiments for stationary methods have been given in Chap. 4. The computations have been performed on an SGI Origin 2000 machine, with four parallel 195 MHZ parallel processors, 512 MB RAM and 4MB cache. There is no parallel component in the implementation; the actual computations are performed by a single processor on the machine. The computer programs are written in Matlab 5.

#### Test Problem 1

Consider the separable problem

\[-\Delta u + p_1 x u_x + p_2 y u_y + p_3 z u_z = w(x, y, z)\]  \hspace{1cm} (5.12)

on \(\Omega = (0,1) \times (0,1) \times (0,1)\), with zero Dirichlet boundary conditions, where \(w(x, y, z)\) is constructed so that the solution is

\[u(x, y, z) = x y z (1 - x) (1 - y) (1 - z) \exp(x + y + z).\]  \hspace{1cm} (5.13)

First some convergence analysis for block stationary methods is provided. For notational convenience, denote \(\gamma = \frac{p_1 h}{2}, \delta = \frac{p_2 h}{2}\) and \(\mu = \frac{p_3 h}{2}\). Suppose \(h\) is sufficiently small and centered difference discretization is performed. Below we refer to the components of the computational molecule, as explained in Chap. 4. Since the problem is separable, each of the components of the computational molecule depends only on one variable, and thus can be written with a single subscript. The components in the \(x\)-direction satisfy

\[c_{i+1}d_i = (1 + \gamma x_{i+1})(1 - \gamma x_i) = 1 + \gamma h - \gamma^2 h^2(i^2 + i),\]
which leads to
\[ 1 + \gamma h - \gamma^2 (1 - h) \leq c_{i+1} d_i \leq 1 + \gamma h - 2\gamma^2 h^2. \]
If \(-1 < \gamma < \frac{1}{2h}\), then \(c_{i+1} d_i > 0\). For \(b_{j+1} c_j\) and \(f_{k+1} g_k\), the bounds are obtained in an identical manner. The center of the computational molecule is exactly \(a = 6\). In terms of the PDE coefficients, the condition on \(\gamma\) means that the convergence analysis performed in Chap. 4 for centered difference discretization of the convective terms is applicable if the PDE coefficients are \(O(n)\). In this case the matrix is symmetrizable by a real diagonal matrix. Using the notation of Chap. 4, let \(\hat{S}\) be the symmetrized matrix, let \(\beta_x = 1 + \gamma h - 2\gamma^2 h^2\), \(\beta_y = 1 + \delta h - 2\delta^2 h^2\), \(\beta_z = 1 + \mu h - 2\mu^2 h^2\), and let \(S^*\) be a modified version of \(\hat{S}\), such that each occurrence of \(c_{i+1} d_i\), \(b_{j+1} c_j\) or \(f_{k+1} g_k\) in \(\hat{S}\) is replaced by the bounds, namely \(\beta_x\), \(\beta_y\) and \(\beta_z\) respectively. Since \(S^* \geq \hat{S}\), \(S^*\) is a symmetrized version of a matrix corresponding to the constant coefficient case, and by Cor. 2.5 is a diagonally dominant \(M\)-matrix. Using Theorem 4.8, bounds on the convergence rates of block stationary methods are available.

<table>
<thead>
<tr>
<th>splitting</th>
<th>1D</th>
<th>2D</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>n^3</td>
<td>(\rho)</td>
</tr>
<tr>
<td>8</td>
<td>512</td>
<td>0.793</td>
</tr>
<tr>
<td>12</td>
<td>1,728</td>
<td>0.895</td>
</tr>
<tr>
<td>16</td>
<td>4,096</td>
<td>0.937</td>
</tr>
<tr>
<td>20</td>
<td>8,000</td>
<td>0.958</td>
</tr>
<tr>
<td>24</td>
<td>13,824</td>
<td>0.970</td>
</tr>
</tbody>
</table>

Table 5.3: comparison between the computed spectral radii of the block Jacobi iteration matrices and the bounds, using centered differences, for the two splittings, with \(p_1 = p_2 = p_3 = 1\).

As opposed to the constant coefficient case, here the bounds are sensitive to sign, and in general, we find that they are tighter if the convective coefficients are negative. An explanation for this is that the values \(\beta_x\), \(\beta_y\) and \(\beta_z\) are larger than 1 if \(p_1\), \(p_2\) and \(p_3\) are positive. On the other hand, if the latter are negative, then we obtain values of the type \(1 - c_1 h^2 - c_2 h^4\) for \(\beta_x\), \(\beta_y\) and \(\beta_z\), with \(c_1, c_2 > 0\). These values are closer to the values that would be obtained
Chapter 5. Solvers, Preconditioners, Implementation and Performance

if the analogous constant coefficient problem was considered, for which the bounds have been shown in Chap. 4 to be very tight. In Table 5.3 the tightness of the bounds is demonstrated; in this case their quality is as good as bounds for the constant coefficient case. In fairness we remark that for larger values of the PDE coefficients the tightness of the bounds deteriorates. An explanation for this is that the values $\beta_{x_1}$, $\beta_y$ and $\beta_z$, which are attained for $x = h$, $y = h$ and $z = h$, are very loose bounds for gridpoints whose coordinate values are close to 1.

In Table 5.4 the performance of solvers for the reduced and the unreduced systems for $p_1 = 50$, $p_2 = 20$, $p_3 = 10$ is compared. $\|r^{(i)}\| < 10^{-10}$ was used as a stopping criterion, where $r^{(i)}$ is the residual at the $i$th iterate. The method that is used is Bi-CGSTAB, preconditioned by ILU(0). In this example Bi-CGSTAB was implemented using Netlib’s Matlab routine. The rate of increase in iteration count as the grid is refined is in agreement with theory, at least if one makes the assumption that for this well conditioned and mildly nonsymmetric system, the convergence behavior is qualitatively the same as that of the conjugate gradient method for symmetric problems of the same type. The preconditioned matrix in this example has a condition number whose square root is of magnitude $O(h^{-1})$, thus it is expected that refining the mesh by a factor of 2 would result in doubling the iteration count. The factor of time as well as number of floating point operations should be at least 16 as the mesh is refined by a factor of 2, since the iteration count is doubled and the size of the matrix is larger by a factor of 8.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$n^3$</th>
<th>iterations</th>
<th>time (sec.)</th>
<th>Mflops</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>unreduced</td>
<td>reduced</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>------</td>
<td>-------------</td>
<td>-------------</td>
<td>--------</td>
</tr>
<tr>
<td>8</td>
<td>512</td>
<td>11</td>
<td>6</td>
<td>0.26</td>
</tr>
<tr>
<td>16</td>
<td>4,096</td>
<td>22</td>
<td>11</td>
<td>4.49</td>
</tr>
<tr>
<td>32</td>
<td>32,768</td>
<td>45</td>
<td>23</td>
<td>97.3</td>
</tr>
<tr>
<td>64</td>
<td>262,144</td>
<td>99</td>
<td>44</td>
<td>2,499</td>
</tr>
</tbody>
</table>

Table 5.4: comparison of solving work/time of the unreduced and the reduced solvers for various mesh sizes, using preconditioned Bi-CGSTAB, for $p_1 = 50$, $p_2 = 20$, $p_3 = 10$. 

130
Chapter 5. Solvers, Preconditioners, Implementation and Performance

The solver of close to symmetric systems converges slower than the solver of more strongly nonsymmetric systems: a solver applied to \( p_1 = p_2 = p_3 = 1 \) on a 64 \( \times \) 64 \( \times \) 64 grid (262,144 unknowns) converges in 173 iterations, and the CPU time is 4,390 seconds. The convergence is thus slower than the convergence for larger magnitudes of the PDE coefficients and the same grid size - compare to the last row in Table 5.4. Fig. 5.6 presents the norm of the relative residual throughout the iteration for the case which is closer to symmetric.

![Figure 5.6: \( l_2 \)-norm of relative residual for preconditioned Bi-CGSTAB applied to a 262,144 \( \times \) 262,144 system corresponding to discretization of the test problem with \( p_1 = p_2 = p_3 = 1 \), on a 64 \( \times \) 64 \( \times \) 64 grid.]

In Table 5.5 estimates of the condition numbers of the reduced and unreduced matrices for \( p_1 = 500 \), \( p_2 = 200 \) and \( p_3 = 100 \) with upwind difference discretization are presented. The estimates were obtained using Matlab's command 'condest'.

Next, we discuss the question which method should be used. As far as Krylov subspace solvers are concerned, it has been shown in [82] for three methods (CGN, CGS, GMRES), that for each of them there can be found an example where it fails and an example where it is superior to the other methods. For our reduced system we have no knowledge of how to give analytical justification for preferring one method over another, and we have to settle for numerical experiments for particular cases and/or follow "recipe" recommendations (e.g. the
Chapter 5. Solvers, Preconditioners, Implementation and Performance

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\kappa_2(U)$</th>
<th>$\kappa_2(R)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>420.85</td>
<td>195.04</td>
</tr>
<tr>
<td>12</td>
<td>1,049.0</td>
<td>489.83</td>
</tr>
<tr>
<td>16</td>
<td>1,859.4</td>
<td>866.53</td>
</tr>
<tr>
<td>20</td>
<td>2,693.7</td>
<td>1,258.6</td>
</tr>
<tr>
<td>24</td>
<td>3,578.4</td>
<td>1,657.1</td>
</tr>
</tbody>
</table>

Table 5.5: comparison between estimates of condition numbers of the unreduced matrix (denoted by 'U') vs. the reduced matrix ('R'), for $p_1 = 500$, $p_2 = 200$, $p_3 = 100$.

“flowchart” given in [11] for picking a solver). Consider the case $p_1 = p_2 = p_3 = 10$. That is,

$$-\varepsilon \Delta u + (x, y, z)^T \nabla u = f,$$

(5.14)

with $\varepsilon = \frac{1}{10}$. Zero is used as an initial guess, and centered difference discretization is performed on a $16 \times 16 \times 16$ grid. In Table 5.6 we examine the performance of block stationary methods. The convergence criterion was $||r^{(i)}||_2/||r^{(0)}||_2 < 10^{-10}$. The values for 'Mflops' have been obtained by using Matlab’s command 'flops'. For all cases Matlab’s builtin functions have been used.

<table>
<thead>
<tr>
<th>method</th>
<th>time (sec.)</th>
<th>iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jacobi</td>
<td>16.02</td>
<td>305</td>
</tr>
<tr>
<td>Gauss-Seidel</td>
<td>11.56</td>
<td>152</td>
</tr>
<tr>
<td>SOR ($\omega = 1.2$)</td>
<td>7.98</td>
<td>97</td>
</tr>
<tr>
<td>SOR ($\omega = 1.4$)</td>
<td>4.38</td>
<td>53</td>
</tr>
<tr>
<td>SOR ($\omega = 1.5$)</td>
<td>2.81</td>
<td>34</td>
</tr>
<tr>
<td>SOR ($\omega = 1.6$)</td>
<td>3.80</td>
<td>46</td>
</tr>
<tr>
<td>SOR ($\omega = 1.8$)</td>
<td>8.76</td>
<td>106</td>
</tr>
</tbody>
</table>

Table 5.6: performance of block stationary methods for Test Problem 1.

From Table 5.6 we can conclude that, as expected, Jacobi and Gauss-Seidel are slow to converge. As for SOR, the optimal relaxation parameter is approximately $\omega^* = 1.5$, and for
this value the performance of the scheme is very good. However, in this case our bound fails to provide an effective approximation to the optimal relaxation parameter. Here we used our bound for the block Jacobi scheme, and the estimate we obtained for $\omega^*$ was 1.851, which is obviously far from the actual optimal relaxation parameter.

Moving to consider preconditioned Krylov subspace solvers, the preconditioners that were used were ILU(0), ILU with numerical dropping (ND), block (1D) Jacobi and SSOR.

<table>
<thead>
<tr>
<th>precondi oner</th>
<th>time (sec.)</th>
<th>Mflops</th>
<th>$\frac{nz[L+U]}{nz[S]}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILU(0)</td>
<td>14.1</td>
<td>0.42</td>
<td>1</td>
</tr>
<tr>
<td>ND(0.1)</td>
<td>0.12</td>
<td>0.074</td>
<td>0.07</td>
</tr>
<tr>
<td>ND(0.01)</td>
<td>2.04</td>
<td>5.47</td>
<td>1.28</td>
</tr>
<tr>
<td>ND(0.001)</td>
<td>5.94</td>
<td>22.29</td>
<td>3.68</td>
</tr>
</tbody>
</table>

Table 5.7: construction time/work of ILU preconditioners. 'ND' stands for numerical dropping, and in brackets the threshold value is given.

The work involved in setting up block Jacobi and SSOR is negligible; for the ILU factorizations Table 5.7 presents the construction time and work (in megaflops), as well as the ratio between the number of nonzeros that were generated in the factors and the number of nonzeros of the coefficient matrix. As was mentioned earlier, the latter is useful in order to estimate the amount of work involved in the preconditioning solve.

The results in Table 5.7 are difficult to interpret. In particular, the construction time of the ILU(0) factorization is extremely long, but the flop count is very small. It seems that this contradiction has to do with Matlab's implementation of the 'luinc' command, and for the purpose of computing overall work, it will be reasonable to take the flop count as a more reliable datum - the number of flops is in accordance with the estimate specified in Eq. (5.8).

In Tables 5.8-5.11 we present our results when testing the various preconditioners with various Krylov subspace solvers. In the Krylov subspace methods considered (BiCG, QMR, Bi-CGSTAB, CGS), two matrix-vector products and two preconditioner solves are required in
Table 5.8: performance of QMR for Test Problem 1.

<table>
<thead>
<tr>
<th>preconditioner</th>
<th>time (sec.)</th>
<th>Mflops</th>
<th>iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILU(0)</td>
<td>3.77</td>
<td>9.8</td>
<td>22</td>
</tr>
<tr>
<td>ND(0.1)</td>
<td>6.05</td>
<td>15.28</td>
<td>49</td>
</tr>
<tr>
<td>ND(0.01)</td>
<td>3.28</td>
<td>8.77</td>
<td>18</td>
</tr>
<tr>
<td>ND(0.001)</td>
<td>3.22</td>
<td>9.23</td>
<td>11</td>
</tr>
<tr>
<td>BJ</td>
<td>7.08</td>
<td>18.22</td>
<td>49</td>
</tr>
<tr>
<td>SGS</td>
<td>4.55</td>
<td>11.99</td>
<td>27</td>
</tr>
</tbody>
</table>

Table 5.9: performance of BiCG for Test Problem 1.

<table>
<thead>
<tr>
<th>preconditioner</th>
<th>time (sec.)</th>
<th>Mflops</th>
<th>iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILU(0)</td>
<td>3.51</td>
<td>8.9</td>
<td>22</td>
</tr>
<tr>
<td>ND(0.1)</td>
<td>5.82</td>
<td>13.99</td>
<td>51</td>
</tr>
<tr>
<td>ND(0.01)</td>
<td>3.06</td>
<td>7.99</td>
<td>18</td>
</tr>
<tr>
<td>ND(0.001)</td>
<td>3.03</td>
<td>8.55</td>
<td>11</td>
</tr>
<tr>
<td>BJ</td>
<td>6.56</td>
<td>16.35</td>
<td>49</td>
</tr>
<tr>
<td>SGS</td>
<td>4.26</td>
<td>10.91</td>
<td>27</td>
</tr>
</tbody>
</table>

each iteration. For Bi-CGSTAB, convergence can occur after ”half an iteration” [11]. We also remark that restarted GMRES was also tested and was generally slower to converge.

In general, Bi-CGSTAB and CGS are more efficient than BiCG and QMR. In fact, the latter are slower by a significant factor. We have no simple explanation for this; note that BiCG and QMR involve computing the transpose of the matrix, which in our implementation and for this problem is available, but might not always be available. CGS and Bi-CGSTAB are on the same level of efficiency for this problem, with marginal difference in performance (except for an unclear difference between their performance when block Jacobi preconditioning is used). Of the preconditioners, ILU(0) seems to be most effective in this case. We remark that when testing problems in strong convection-dominated regions, we did find examples where
Chapter 5. Solvers, Preconditioners, Implementation and Performance

<table>
<thead>
<tr>
<th>Preconditioner</th>
<th>Time (sec)</th>
<th>Mflops</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILU(0)</td>
<td>1.8</td>
<td>6.01</td>
<td>12</td>
</tr>
<tr>
<td>ND(0.1)</td>
<td>3.53</td>
<td>11.07</td>
<td>30</td>
</tr>
<tr>
<td>ND(0.01)</td>
<td>1.51</td>
<td>5.15</td>
<td>9 1/2</td>
</tr>
<tr>
<td>ND(0.001)</td>
<td>1.26</td>
<td>4.83</td>
<td>5 1/2</td>
</tr>
<tr>
<td>BJ</td>
<td>3.61</td>
<td>11.79</td>
<td>27 1/2</td>
</tr>
<tr>
<td>SGS</td>
<td>2.17</td>
<td>7.25</td>
<td>14 1/2</td>
</tr>
</tbody>
</table>

Table 5.10: performance of Bi-CGSTAB for Test Problem 1.

<table>
<thead>
<tr>
<th>Preconditioner</th>
<th>Time (sec)</th>
<th>Mflops</th>
<th>Iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILU(0)</td>
<td>1.74</td>
<td>5.30</td>
<td>13</td>
</tr>
<tr>
<td>ND(0.1)</td>
<td>3.56</td>
<td>10.33</td>
<td>29</td>
</tr>
<tr>
<td>ND(0.01)</td>
<td>1.51</td>
<td>4.91</td>
<td>11</td>
</tr>
<tr>
<td>ND(0.001)</td>
<td>1.23</td>
<td>4.66</td>
<td>6</td>
</tr>
<tr>
<td>BJ</td>
<td>4.31</td>
<td>13.20</td>
<td>37</td>
</tr>
<tr>
<td>SGS</td>
<td>1.90</td>
<td>6.11</td>
<td>15</td>
</tr>
</tbody>
</table>

Table 5.11: performance of CGS for Test Problem 1.

ILU(0) preconditioned solvers performed poorly. SGS is also competitive, in particular because it involves a small set-up time.

Finally, we mention that in cases where the optimal relaxation parameter (or a reasonable approximation to it) is available for the SOR scheme, it performs very well and is highly competitive with preconditioned Krylov subspace solvers. If the bounds presented in Chap. 4 are tight, a good approximation to the optimal relaxation parameter can be obtained. SOR requires only one matrix-vector product per iteration, and is thus significantly cheaper than an iteration of preconditioned Krylov subspace solvers. As an example, consider the same problem with PDE coefficients $p_1$, $p_2$, $p_3 = 1$, for which the optimal relaxation parameter is approximately $\omega^* = 1.5$. For this value the SOR solver converges within 37 iterations. However,
the convergence analysis provided an approximation $\omega = 1.599$; for this value the SOR scheme converges within 46 iterations, and for a fair comparison, this is the datum that should be used. In comparison, ILU(0) preconditioned CGS converges in this case within 11 iterations and Bi-CGSTAB converges within $10\frac{1}{2}$ iterations. Thus SOR and these two Krylov subspace solvers converge within almost the same number of matrix-vector products. If the construction work of the preconditioner is added, the conclusion is that SOR is faster. When the grid is finer, the estimate of the optimal relaxation parameter is more accurate and SOR performs even better. In addition, SOR requires much less storage.

Test Problem 2

Consider the following singularly perturbed problem:

$$-\varepsilon \Delta u + \nabla v \cdot \nabla u = f$$

(5.15)

where

$$v = \frac{1}{2}(x^2 + y^2 + z^2),$$

(5.16)

subject to nonzero Dirichlet type boundary conditions on a square domain $(a_x, b_x) \times (a_y, b_y) \times (a_z, b_z)$ which contains the origin. This class of problems is currently being studied by Sun & Ward - see [103] and references therein.

Eq. (5.15) is similar to the equation considered in Test Problem 1, but now there are turning points inside the domain, which turn the problem into a much more difficult one. The problem comes from a large variety of applications, including semiconductors, population growth [75], the exit problem [77], and more. One interesting phenomenon here is that for any number of space dimensions the continuous problem has an isolated exponentially small eigenvalue. The eigenvalue problem for a class of one-dimensional problems which contain the one-dimensional version of Eq. (5.15) has been studied by De Groen [30]. Asymptotic analysis (for different aspects) has been done by Ludwig [75], Grasman & Mantkowsky [60], Mantkowsky & Schuss [77], and others (see [103] for a list of references). The analysis in these papers shows that the analytical solution of (5.15) has boundary layers near the edges and is constant in the interior.
of the domain. In [60] a variational approach is used to determine the constant as a weighted average of the boundary data.

Finite difference schemes have truncation errors which might be larger than \( \varepsilon \), and it is not clear how accurate the computed solution is. Adjerid, Aiffa & Flaherty [1] have experimented with a similar two-dimensional problem (with different boundary conditions) and used a finite element code. They have observed that iterative solvers converge quickly to a constant in the interior of the domain, but not necessarily to the exact constant predicted by Grasman & Mankowsky’s asymptotic analysis.

Following Sun’s suggestions [102], Il’in scheme was used [74]. Essentially it is a centered difference scheme, modified by using fitting parameters associated with the functions \( \coth x \), \( \coth y \), \( \coth z \) applied to the gridpoints. When \( \varepsilon \) is very small, the scheme behaves like the upwind scheme; when \( \varepsilon \) is large, the scheme is second order accurate [74] (see also [97] for useful analysis and results on the accuracy of finite difference schemes applied to singularly perturbed problems). Eq. (5.15) has been considered with boundary conditions \( u_b = x + y + z \).

The leading order of the asymptotic expansion requires in general computing a multidimensional integral (see [60, p. 594] for the closed form). For the particular PDE coefficients in (5.15) the constant can be obtained by a weighted sum of the contact points of the function \( v \) with the boundary, that are closest to the origin. There are six contact points in this case, all of which are within distance 1 from the origin; the constant in this case is expected to be zero.

The cyclically reduced operator inherits the properties of the continuous problem: for \( \varepsilon = 0.03 \) and discretization on a \( 16 \times 16 \times 16 \) grid, by Matlab’s command ‘eigs’ the three smallest eigenvalues of the reduced matrix (multiplicity excluded) are: \( 2.64 \cdot 10^{-6}, 1.93 \) and 3.59. For this case the condition number estimated by Matlab’s ‘condest’ command is approximately \( 10^9 \).

Table 5.12 presents flop counts and the numerical solution for a problem discretized on a \( 16 \times 16 \times 16 \) grid, using ILU(0) preconditioned Bi-CGSTAB. The numbers represent averages of 5 tests, started with a random initial guess. The sum of all the absolute values of the solution in interior points of the grid, presented in the third column, can be considered an approximation
to the constant in the interior. For $\varepsilon = 0.03$ and $\varepsilon = 0.02$ the accuracy is satisfactory. The average (third column) for $\varepsilon = 0.03$ is larger because the boundary layer is less “sharp” in this case. For $\varepsilon = 0.01$ there is an error in the solution. Here $\varepsilon$ is already significantly smaller than $h$, and in the numerical tests there were fluctuations in the constants obtained, as well as the computational work.

Differences in accuracy between the reduced and the unreduced systems have been observed to be negligible. For this class of problems the reduced solver does not converge significantly faster than the unreduced solver, in particular when $\varepsilon$ gets smaller.

\[
\begin{array}{|c|c|c|}
\hline
\varepsilon & \text{Mflops} & \frac{1}{(n-2)^3} \sum_{i,j,k \neq i,n} |u_{i,j,k}| \\
\hline
0.03 & 10.0 & 7.54e-4 \\
0.02 & 11.1 & 2.71e-5 \\
0.01 & 14.6 & 0.53 \\
\hline
\end{array}
\]

Table 5.12: Test Problem 2: overall flop counts and average computed constants for three values of $\varepsilon$, when ILU(0) preconditioned Bi-CGSTAB is used.

For severely ill-conditioned problems ILU(0) preconditioned iterations do not at all converge. Here the only preconditioners that have been found effective are incomplete LU factorizations with a very small drop threshold. Table 5.13 presents details on numerical experiments for $\varepsilon = 0.005$. Notice the large amount of computational work that is required. The ratio between the number of nonzeros in the factors and the number of the nonzeros of the matrix is given in the column titled \[\frac{\text{nz}(L+U)}{\text{nz}(S)}\]. For comparison, the same ratio for the complete LU factorization is 27.93 in this case. Since $\varepsilon$ is very small, in this discretization the constant in the interior cannot be expected to be computed accurately. We note, though, that the solution for numerical dropping with both $10^{-4}$ and $10^{-6}$ has been observed to be constant in the interior and thus behaves qualitatively like the leading ordering of the analytical solution. This example demonstrates the robustness of incomplete factorizations based on dropping. The computational work that is required is large, but these are the only preconditioners for which the solver converges. Notice also that once these preconditioners obtain convergence, the tradeoff between the larger work
per iteration and the small iteration count translates into almost the same amount of overall computational work (last two rows in the table).

<table>
<thead>
<tr>
<th>method</th>
<th>( nz(L+U) )</th>
<th>Mflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILU(0)</td>
<td>1</td>
<td>N/C</td>
</tr>
<tr>
<td>ND(10^{-2})</td>
<td>1.43</td>
<td>N/C</td>
</tr>
<tr>
<td>ND(10^{-4})</td>
<td>3.25</td>
<td>36.2</td>
</tr>
<tr>
<td>ND(10^{-6})</td>
<td>5.09</td>
<td>40.4</td>
</tr>
</tbody>
</table>

Table 5.13: comparison of performance of various incomplete factorizations in conjunction with CGS, for Test Problem 2. 'N/C' stands for no convergence.

Test Problem 3

Consider the following problem:

\[-\Delta u + v \cdot \nabla u = w, \quad (5.17)\]

on the unit cube, where

\[v = (\sin(\pi y), \sin(\pi x), 1)^T, \quad (5.18)\]

subject to Neumann boundary conditions on \( z = 0, 1 \) and Dirichlet boundary conditions on the rest of the boundary. This is an example of a nonseparable problem, for which the convergence analysis of Chap. 4 does not apply. The Neumann conditions on \( z \)-direction are discretized using centered difference schemes, by adding two artificial planes of variables (whose \( z \) indices are \(-1 \) and \( n + 1 \)). In order to examine the accuracy of the scheme, \( w \) has been constructed so that the analytical solution is

\[u(x, y, z) = \sin(\pi x) \cdot \sin(\pi y) \cdot \cos(\pi z).\]

Table 5.14 presents the nice behavior of the norm of the error, and illustrates the second order accuracy of the scheme.

Table 5.15 compares the performance of the reduced solver to the performance of the unreduced solver, using ILU(0) preconditioned Bi-CGSTAB. The criterion used here is number of flops. All the components are included: construction of the system, construction of the preconditioners, solution and back substitution for the gridpoints which were previously eliminated by
Chapter 5. Solvers, Preconditioners, Implementation and Performance

Figure 5.7: a 2D slice of the numerical solution of Test Problem 3, for $z = \frac{\pi}{17}$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$\frac{|e|}{\sqrt{N}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0.004346</td>
</tr>
<tr>
<td>16</td>
<td>0.001123</td>
</tr>
<tr>
<td>32</td>
<td>0.000285</td>
</tr>
</tbody>
</table>

Table 5.14: norm of the error for Test Problem 3, for various sizes of the grid. In the header, $e$ stands for the error and $N$ stands for the number of unknowns.

the reduction. The performance of the reduced solver is better, and by a larger margin as the grid gets finer. This reflects the fact that the overhead of constructing the cyclically reduced matrix becomes less significant compared to the solve time as the number of unknowns increases. The numbers in the last row (for a $32 \times 32 \times 32$ grid) are typical of the factor of improvement; here the solving time is by far longer than the construction; the ratio of improvement is expected to be similar for larger grids.

5.5 Vector and Parallel Implementation

Since the reduced matrix has a clear block structure, parallel implementation is an attractive option for both constructing the matrix and computing the solution. The need for parallelism is important especially because systems of equations associated with three-dimensional problems are typically very large. For example, a grid of size $64 \times 64 \times 64$ on the unit cube, which might be too coarse for difficult applications, requires solving a non-small system of more than 250,000
Table 5.15: comparison between the performance of the unreduced and the reduced solvers for various grids, using ILU(0) preconditioned Bi-CGSTAB, for Test Problem 3.

<table>
<thead>
<tr>
<th>( n )</th>
<th>( n^3 )</th>
<th>Mflops</th>
</tr>
</thead>
<tbody>
<tr>
<td>---</td>
<td>---</td>
<td>( \text{unreduced} )</td>
</tr>
<tr>
<td>8</td>
<td>512</td>
<td>1.17</td>
</tr>
<tr>
<td>16</td>
<td>4,096</td>
<td>14.03</td>
</tr>
<tr>
<td>32</td>
<td>32,768</td>
<td>237.83</td>
</tr>
</tbody>
</table>

unknowns. The problem here is storage as well as time. If the matrix is too big to fit in the physical memory of the computer, swapping (paging) takes place, which substantially slows the execution time (the computation time could slow down by a factor of 50 [87]). This section addresses a few issues associated with parallel and vector implementation. It should be viewed only as a general overview; in practice our implementation is vectorized, written in Matlab, and includes no parallel components.

Overview

General overviews on parallel implementation of solution of linear systems are, for example, Demmel, Heath & van der Vorst [32], and an earlier review by Ortega & Voigt [84]. The earliest forms of parallelism (or vectorization, rather) were based on having a single processor with multiple functional units, such as multipliers and adders. During compilation, dependence analysis was done, and so, if for example a certain computation involved four additions and two multiplications which were independent of one another, these operations were performed simultaneously using four adders and two multipliers. The next stage was pipelining. Golub and Van Loan [55] illustrate pipelining by making an analogy to an assembly line used in car manufacturing. Suppose a certain operation can be divided into \( m \) stages, each of which takes \( t \) seconds to complete. If we have \( n \) such operations to perform, then the overall time it would take when performing all the operations sequentially is \( m \cdot n \cdot t \). If instead of waiting for the first operation to be fully completed before performing the second operation we perform the first stage of operation number 2 while the second stage of operation number 1 is taking place, and
so on, the overall amount of time is the necessary time $m \cdot t$, plus the overhead of the waiting time for operation number $n$ to start, which is $(n - 1) \cdot t$. This is a substantial improvement over the time it takes to perform the computation sequentially. An important term here is \textit{speed-up}, which is the time it takes to perform an algorithm sequentially, divided by the time it takes to perform it in the parallel environment the user has. In this particular example we have speed-up of $\frac{mn}{m+n-1}$, which is close to $m$ for $n$ sufficiently large.

The first vector computers used pipelined functional units, and had the capability to work very efficiently on vectors and arrays by using vector instructions as part of their instruction sets. Such instructions included loading and storing vectors, adding or subtracting vectors, performing dot products and so on. Users were not required to specify their desire to perform these operations in a vectorized manner; rather, the object-code for pipelining was typically generated in compilation time, and it was the role of the compiler to identify components that could be pipelined.

Today’s parallel computations are based on multiprocessor systems, with several processors, each having a CPU, local memory, and capability to pass messages to other computers in the network. Loosely speaking, there are two main philosophies as to the architecture of a parallel network [32]:

1. Shared memory architecture.

2. Distributed memory architecture.

Shared memory architectures are ones where each processor has access to a global, common memory area. The advantage of this architecture is that as far as the user is concerned, any processor can access any data, and very little work has to be done in order to manage data access. This makes the programming considerably easy. On the other hand, locality of data (as is the case with discretized partial differential equations) cannot be taken advantage of as efficiently as in other architectures [95].

In a distributed memory environment a large number of processors are connected in some topological form; each of them has its own local memory, and executes its own programs. Data
Chapter 5. Solvers, Preconditioners, Implementation and Performance

is sent to other processors in the network in the form of messages. The network topology defines how the processors are interconnected. Examples for possible topologies are rings, meshes, hypercubes and trees. Description of these architectures can be found, e.g., in [95],[55]. Meshes of processors well suit solution of discretized partial differential equations, as groups of gridpoints can be straightforwardly mapped into each of the processors. A 2D mesh is illustrated in Fig. 5.5.

![Figure 5.8: 2D mesh architecture (3 x 3).](image)

When considering solving a linear system on a parallel machine using preconditioned Krylov subspace methods, different aspects need to be addressed: preconditioner set-up, matrix-vector multiplication, vector updates, dot products and preconditioner solve. For a general overview, see [87]. The preconditioner set-up and the preconditioner solve are usually the bottleneck. Dot products are easy to handle, but require a fairly large communication, as each processor performs the computation for its assigned variables, but the final result is required by all the processors.

**Construction of the reduced system**

For the reduced system, when considering distributed architecture, for simplicity it is convenient to refer to a 3D mesh of \( p^3 \) processors, assume that we have a uniform mesh with \( n^3 \) unknowns, and that \( p \) divides \( n \) (if this is not so, then we can proceed by mapping an unequal number of gridpoints into each of the processors).

One important advantage of the ordering strategies discussed in Chap. 3 is that all the block rows have the same size, and thus the issue of load balancing [32],[55],[87] in this particular
aspect is resolved. Our working assumption is standard [95]: pairs of rows-unknowns are handled by the same processor. That is, row number $i$ and unknown number $i$ are both mapped into the same processor. Usually for PDEs, due to the locality of the operators, the geometry of the gridpoints is the source for mapping the unknowns into the processors in the network. We can thus assign subcubes of the original domain to each processor. As a result, each processor constructs a part of the reduced matrix which is a rectangular matrix whose number of rows is the number of unknowns that are associated with it, and the number of columns is the size of the reduced matrix. If only a 2D mesh of processors is available, then mapping of the unknowns can be done by assigning “stripes”, for example of size $n \times 2 \times 2$ (equivalent to the 1D sets of the two-plane ordering, described in Chap. 3) to each of the processors. However, the drawback here is that it allows less flexibility than a 3D mesh of processors in dynamically changing the orientation of the blocks relative to the independent variables $x$, $y$ and $z$.

The construction of the reduced matrix entries is divided into two stages:

1. Compute the components of the seven-point operator.

2. Perform the algebraic reduction.

The first stage can be parallelized in a straightforward manner. For the reduction step, constructing a row associated with a certain gridpoint, the values of the computational molecule of its neighbors are required. If $(i, j, k)$ is the point being considered, then its furthest neighbors are $(i \pm 2, j, k), (i, j \pm 2, k), (i, j, k \pm 2)$; the rest of the neighbors are within a smaller distance of $\sqrt{2}$. Each processor must have data on an external interface layer of thickness 2 on each of the six sides of the subcube (see Fig. 5.9). There are two ways to obtain the values of the computational molecule associated with the gridpoints that belong to the external interface layer: either by communicating with neighboring processors, or by computing these values as part of the first stage, which will cause overhead in arithmetic. Since typically the number of gridpoints in the mesh is very large, and since communication will require message exchange with a large number of 26 processors, the second option (of overlap in arithmetic) seems a better alternative as it eliminates communication.
The computation of the entries of the reduced matrix is thus almost fully parallelizable, with minimum communication and overhead caused by additional computation of components of the computational molecule for external interface variables. Suppose computing a value of the computational molecule of the seven-point operator takes $f_1$ flops and computing each component for the reduced computational molecule takes $f_2$ flops. Then, for step 1, the amount of computations per each processor (referring to processors associated with an interior subdomain) is $(\frac{n+1}{p})^3 \cdot f_1$ flops, and for step 2 the work involved is $\frac{1}{2} \cdot \left(\frac{n}{p}\right)^3 \cdot f_2$ flops. Thus the speed-up in arithmetic is nearly $p$ if $n \gg p$.

In a vector environment the computation of the components of the matrix is vectorizable. Here both construction diagonal by diagonal or construction block by block can be efficiently vectorized. When constructing diagonal by diagonal, the idea is to reshape the three-dimensional arrays which contain the previously computed values of the computational molecule of the cyclically reduced operator (which can clearly be done in a vectorized manner) into a one-dimensional array, and correct to zero the values that correspond to values of the computational molecule which are zero due to the associated grid point being close to the boundary. In an environment which has a parallel as well as a vector component, the construction of all the diagonals of the matrix can be done in parallel, provided that all the processors have copies of the values of the computational molecule of the whole domain (here, a shared memory
environment might be appropriate). As an example, below we present a piece of code written in Matlab syntax which assigns two (arbitrarily picked) diagonals of the reduced matrix which corresponds to natural lexicographic ordering.  

\[
dzmm = -(b(i,j,k-1)*f(i,j,k)/a(i,j,k-1) ... \\
    + b(i,j,k)*f(i,j-1,k)/a(i,j-1,k)) \\
dzmm(:,1,:) = 0; dzmm(:,:,1) = 0; \\
diag_zmm = reshape(dzmm,sys_size,1); \\
zmm = lny*lnx*lny; \\
diag_zmm(1:sys_size-izmm) = diag_zmm(izmm+1:sys_size); \\
dppz = -(d(i,j+1,k)*e(i,j,k)/a(i,j+1,k) ... \\
    + d(i,j,k)*e(i+1,j,k)/a(i+1,j,k)) \\
dppz(lnx,:, :) = 0; dppz(:,lny,:) = 0; \\
diag_ppz = reshape(dppz,sys_size,1); \\
ippz = lny+1; \\
diag_ppz(ippz+1:sys_size) = diag_ppz(1:sys_size-ippz); \\
\]

In the above code, \( a, b, d, e, f \) are three-dimensional arrays which hold the previously computed components of the computational molecule, \( lnx, lny, lnz \) are the numbers of components in each direction, \( diag_zmm \) is the diagonal that contains gridpoints whose coordinates are indexed \((i,j-1,k-1)\) and \( diag_ppz \) corresponds to the gridpoints indexed \((i+1,j+1,k)\). The integer numbers \( izmm \) and \( ippz \) are the indices of the diagonals; finally, the values are shifted so as to fit in the matrix. The shift is different between superdiagonals and subdiagonals.

Reordering is done by using vectors which map coordinates to row \# in the matrix and vice versa. For example, for the two-plane ordering, the connection between the gridpoints location in the matrix \( \ell \) and its coordinate values is given by

---

1In the actual Matlab program the implementation is different. The code presented here is only for the purpose of illustration of the general idea.
Chapter 5. Solvers, Preconditioners, Implementation and Performance

\[ i = \text{fix}\left\{\left(\ell - 1\right) \mod (2n)\right\}/2 + 1 \]  \hspace{1cm} (5.19a)

\[ j = \begin{cases} 2 \cdot \text{fix}\left(\frac{\ell - 1}{n}\right) + 1 & \ell \mod 4 = 0 \text{ or } 1 \\ 2 \cdot \text{fix}\left(\frac{\ell - 1}{n^2}\right) + 1 & \ell \mod 4 = 2 \text{ or } 3 \end{cases} \]  \hspace{1cm} (5.19b)

\[ k = \begin{cases} 2 \cdot \text{fix}\left(\frac{(\ell - 1) \mod n^2}{2n}\right) + 1 & \ell \text{ odd} \\ 2 \cdot \text{fix}\left(\frac{(\ell - 1) \mod n^2}{2n}\right) + 1 & \ell \text{ even} \end{cases} \]  \hspace{1cm} (5.19c)

For the two-line ordering, the connection is

\[ i = \left(\ell - 1\right) \mod n + 1, \]  \hspace{1cm} (5.20a)

\[ k = \text{fix}\left(\frac{\ell - 1}{n^2/2}\right) + 1, \]  \hspace{1cm} (5.20b)

\[ j = \begin{cases} 2 \cdot \left(\text{fix}\left(\ell - (k - 1) \cdot (n^2/2) - 1\right)/n\right) + 1 & k \text{ odd} \\ 2 \cdot \left(\text{fix}\left(\ell - (k - 1) \cdot (n^2/2) - 1\right)/n\right) + 2 & k \text{ even} \end{cases} \]  \hspace{1cm} (5.20c)

Matrix-vector products for the reduced matrix

One drawback that the reduced matrix has, compared to the unreduced matrix, is that its associated computational molecule is non-compact. As a result, when performing matrix-vector products each of the processors in the network must communicate with all the processors surrounding it, which amount to 26 for “interior” processors. This can be observed by examining the computational molecule, see Fig. 2.4. However, the messages that need to be exchanged are very small in size and the communication can be done simultaneously with other computations, as will be clarified below.
For the purpose of the illustration, it is convenient to assign two colors to the processors in the network: suppose the processors are either “red” or “black”, by the usual red/black ordering of a 3D mesh (see Fig. 2.1). Also, suppose the processor that is being considered is one which is an interior processor in the mesh of processors, and let $P$ stand for its “id number”. Then the sizes of the messages exchanged by processor $P$ and its neighboring processors in the process of computing the product of the reduced matrix with a vector are:

- $\left(\frac{n}{p}\right)^2$ numbers for the six processors of the opposite color. Here a layer of size $\frac{n}{p} \times \frac{n}{p} \times 2$ of external interface variables is required. It contains $\left(\frac{n}{p}\right)^2$ unknowns, not $2\left(\frac{n}{p}\right)^2$, because only the black gridpoints appear in the reduced grid.

- Either $\frac{n}{p}$ or less numbers for the processors with the same color of processor $P$.

Thus six processors will transfer most of the data required. This is different from the unreduced system where these same six processors will transfer all the required data to processor $P$. The sizes of the messages are small, and as an illustration, consider a network of 1,000 processors with 1,000,000 unknowns; then six processors exchange 100 numbers with processor $P$, and the rest exchange 10 or less numbers.

The local variables that cause the need to exchange messages are the ones close to the boundary: the local interface points [95]. For the reduced system, the thickness of this layer in terms of unknowns is two, just like the thickness of the external interface layer. This follows from the structure of the reduced computational molecule. For the unreduced system the thickness of these layers is one. The local points (see Fig. 5.9) are the points which form the dense “block diagonal” part of the rectangular matrix.

The actual matrix-vector product consists of three separate stages, the first two of which can be done in parallel [95]: multiplication of local variables, message exchange and multiplication of external variables. The size of the each subcube in the reduced grid is $\frac{n}{p}$, and it contains $\frac{1}{2} \cdot (\frac{n}{p})^3$ unknowns; in order to illustrate how this works, we refer below to a particular example: we take $n = 16$, $p = 4$ and refer to natural lexicographic ordering. The interior subcube is indexed $(2, 2, 2)$ in the mesh of processors (the equality of the indices here has no special
meaning whatsoever). The subcube indices in this case, in terms of the index of the *original ordering* of the reduced grid, are: \((547, 548, 555, 556, 563, 564, 571, 572) + i \cdot 128, \ i = 0, 1, 2, 3\); there are \(32 = \frac{4^3}{2}\) gridpoints in total.

The sparsity pattern of the \(32 \times 2048\) rectangular matrix which is the part of the reduced matrix handled by processor \(P\) is depicted in Fig. 5.10. This is *before* the unknowns were locally ordered. In the top figure, the spy pattern of the original reduced (square) matrix illustrates the (two) 2D sets of gridpoints (see Chap. 3 for definition) in which the subcube we are considering is contained; these are 32 of the rows between the two horizontal lines, namely between rows \(512 + 1\) and \(1024\) (indeed, \(1024 - 512 = 2n^2\)). The bottom figure is the sparsity pattern of the rectangular matrix of size \(32 \times 2048\) which contains all the rows associated with the gridpoints of the subcube. As is evident, the matrix contains several nonzero blocks, more dense in some places (which correspond to the location of the *local* gridpoints).

![Figure 5.10](image)

Figure 5.10: the part of the reduced matrix that contains the gridpoints associated with subcube \((2, 2, 2)\), and the rectangular matrix before local ordering in processor \(P\) has been done.

After local ordering of the unknowns is performed (which is done by using the same strategy as used for the whole reduced grid) the result is a rectangular matrix whose "diagonal block" is fairly dense, and corresponds to the *local* variables. Fig. 5.11 illustrates this for our particular example. Here the band or the sparsity pattern of the matrix are not important; what is
important is the density. In this example the number of nonzeros is in the main block is 344, compared to 608 of the whole rectangular matrix (see Fig. 5.10). In practice, for large \( n \), the situation is much better than in this example (where \( n \) is very small), in the sense that most of the nonzeros of the rectangular matrix appear in the main diagonal block and most of the rows will have 19 nonzero elements (equal to the number of components in the reduced computational molecule). For these local points no communication is required in order to perform the matrix-vector product. In order to perform the multiplication of the local part more efficiently, re-ordering of the “block diagonal” matrix could be done, so that all the local points are ordered first, and only then the local interface points appear.

![Figure 5.11: the “local part” of the rectangular matrix depicted in Fig. 5.10, after re-ordering.](image)

If \( n \gg p \), for the reduced system there are approximately \( \frac{1}{2} \cdot \left( \frac{n-1}{p} \right)^3 \) local points and only approximately \( 6 \cdot \left( \frac{n}{p} \right)^2 \) local interface points and about the same number of external points. That is, the external matrix (that is, the rectangular matrix, with the “block diagonal” excluded) has a number of nonzeros which is smaller by an order of magnitude from the number of nonzeros in the square matrix that corresponds to the “block diagonal”. This is important, because it means that the communication required for exchanging values of the external interface variables does not form a bottleneck - it can be done while a massive \( O(n^3) \) computation is being carried out. The communication with the neighboring processors will include also transmitting numbers corresponding to the local interface points; these are considered external interface points for the neighboring processors.

Once the stage where simultaneous computation of the local matrix-vector product and the
exchange of local and external interface variables has been completed, what is left is to perform and add the inexpensive component of the external matrix-vector product. For the reduced matrix this operation takes approximately $12n^2$ floating point operations.

**Preconditioner set-up and solve**

Figure 5.12: a 2D slice of processors in a $7 \times 7 \times 7$ mesh that hold entries of 1D or 2D sets of unknowns that are associated with the unknowns mapped into the processor marked by 'X', or communicate with this processor when matrix-vector products are performed.

It has already been mentioned before, that the solve is the costly part of the computation. One of the most important aspects here is the ordering strategy. In fact, introducing the four-color block ordering (for 1D partitioning) and the red/black block ordering (for 2D partitioning) in Chap. 3, and testing their performance (Chap. 4) is mainly for the purpose of vectorization and parallelization. These ordering strategies are less efficient than natural ordering. Intuitively, this can be easily explained by observing that the number of nonzeros of these orderings in the (complete) LU is larger, and thus an incomplete LU factorization is less accurate. On the other hand multicolor orderings are easily parallelizable. Experimental investigation of effects of coloring on performance of parallel solvers is provided, for example, in [88].

For the reduced system, the four-color scheme (Fig. 3.6) gives rise to solution in parallel for four blocks of unknowns, according to their color. It should be noted, however, that the diagonal blocks are block diagonal matrices; Fig. 5.12 depicts the processors which hold unknowns associated with a 1D or 2D block, or communicate with the processor marked by 'X'.
when matrix-vector products are performed. In a network of \( p^3 \) processors, approximately \( 3p \) processors will hold the entries of a 1D or a 2D set of unknowns of the reduced grid.

Table 5.16 presents iteration counts for ILU(0) preconditioned Bi-CGSTAB, applied to Test Problem 1 (Sec. 5.4) with \( p_1 = p_2 = p_3 = 2 \). The natural ordering is more efficient than multicolor version (the difference in percentage is not negligible) but would obtain much smaller speed-up rates.

Investigation of parallel solution techniques for the reduced system should be subject to further investigation and is left as a possible future project.

<table>
<thead>
<tr>
<th>ordering</th>
<th>iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural</td>
<td>16</td>
</tr>
<tr>
<td>1D four-color</td>
<td>19 ( \frac{1}{2} )</td>
</tr>
<tr>
<td>2D red/black</td>
<td>20 ( \frac{1}{2} )</td>
</tr>
</tbody>
</table>

Table 5.16: comparison of iteration counts of natural and multicolor orderings for Test Problem 1 with \( p_1 = p_2 = p_3 = 2 \).
Chapter 6

Summary, Conclusions and Future Research

In this dissertation a three-dimensional cyclically reduced linear system of equations arising from the steady-state convection-diffusion equation has been derived and analyzed. Analysis of cyclic reduction for non-self-adjoint problems was done for the first time only in the early 1990s, when Elman & Golub considered the two-dimensional problem. Derivation and analysis of the three-dimensional non-self-adjoint problem has been done in this thesis for the first time. A summary of the main results, conclusions and suggestions of future research in related topics is given below. The main results of this thesis are also presented in [62],[63],[64].

6.1 Summary and Conclusions

The following issues have been addressed:

1. Derivation of the linear system for both the constant and the variable coefficient problems. The step of cyclic reduction for three-dimensional problems has been described in detail. General numerical properties of the reduced matrix have been described. It has been shown that the reduced matrix is diagonally dominant, symmetrizable or an $M$-matrix whenever the original matrix is, and moreover, there are regions of PDE coefficients for which only the reduced matrix has such properties. The reduced matrix is generally better conditioned than the original matrix.

2. Orderings. The reduced grid is different than a standard tensor-product grid, and re-ordering of the unknowns is an important issue. A general block ordering strategy for three-dimensional grids has been presented. The approach is based on referring to
one-dimensional or two-dimensional grids of sets of gridpoints and their associated block computational molecule. A highly effective family of two-plane orderings has been presented, and comparisons with other ordering strategies have been performed. Natural as well as multicolor block versions have been considered.

3. **Symmetrization.** In order to obtain bounds on convergence rates of stationary methods, Elman & Golub’s strategy of finding a diagonal symmetrizer has been adopted. Here it turns out that the three-dimensional case is similar to the two-dimensional case: in both cases the reduced matrix can be symmetrized for a substantially larger range of PDE coefficients compared to the original matrix. In particular, the reduced matrices for both the 2D and 3D problems can be symmetrized when all the mesh Reynolds numbers are larger than 1 in magnitude.

4. **Bounds on convergence rates.** Comprehensive convergence analysis has been performed for block stationary methods. First, tight bounds on convergence rates of the block Jacobi scheme have been derived. The convergence analysis applies to upwind difference discretization with any value of the PDE coefficients, or centered difference discretization with mesh Reynolds numbers smaller than 1. Two partitionings based on dimension (1D and 2D) have been considered. The reduced matrix is not consistently ordered relative to 1D partitioning, and in order to use Young’s analysis for Gauss-Seidel and Successive-Over-Relaxation, it has been shown by extensions of the theory for $p$-cyclic matrices, due to Varga, that the two-plane matrix is nearly consistently ordered relative to this partitioning for the range of PDE coefficients for which the convergence analysis applies. The bounds on convergence rates for block Gauss-Seidel are tight, and for SOR the estimate of the optimal relaxation parameter, obtained by using Young’s formula, is accurate. Fourier analysis extends the convergence results to the case of centered difference discretization of convection dominated equations.

5. **Computational work.** Careful analysis of the work involved in each of the solvers has been performed. In particular, when the PDE coefficients are such that the numerical solution is stable and convergence is guaranteed, it has been shown that block solvers based
on 1D partitioning are generally more efficient than solvers based on 2D partitioning.

6. **Preconditioners** have been considered. For incomplete LU factorizations, the amount of fill-in in the reduced matrix and the computational work involved in constructing the preconditioner have been discussed. The construction of incomplete LU factorizations is more costly for the reduced matrix, compared to the unreduced matrix. Experimental results show that when the problem is not too ill conditioned, ILU(0) is more effective than other preconditioners; incomplete factorizations based on threshold require more computational work but perform well for ill-conditioned problems.

7. **The cost of the construction** of the reduced system has been addressed, and it has been shown that the construction is an inexpensive component in the overall computational work required to compute the solution.

8. **Comparison with the unreduced system** has been performed, and it has been shown that reduced solvers typically converge faster. This has been proven analytically for block stationary methods, and has been demonstrated empirically for a variety of Krylov subspace solvers. Also, it has been shown that there is a range of PDE coefficients for which the unreduced solver diverges whereas the reduced solver converges.

9. **Experimental comparison between various solvers** has been performed. For preconditioned Krylov subspace solvers Bi-CGSTAB and CGS seem to be more efficient than other Krylov subspace methods that have been considered. They seem more appropriate to use for other reasons as well - due to the high memory requirements for the class of problems considered in this thesis GMRES might not be the first choice, and due to the need to have the transpose matrix, QMR and BiCG are less preferable.

   When the optimal relaxation parameter is known, which is the case for constant coefficient problems and certain variable coefficient problems, SOR is highly competitive with Krylov subspace solvers. This underlines the importance of the convergence analysis that has been performed in Chapter 4.

10. **Implementation** has been discussed. Some preliminary observations on aspects of vec-
Chaper 6. Summary, Conclusions and Future Research

torization and parallelism have been provided. The construction of the reduced system is fully parallelizable and highly vectorizable; four-color 1D block orderings and red/black 2D block orderings are appropriate for parallel implementation of solvers.

11. Numerical results which validate the analysis and illustrate various aspects have been presented.

The issue of ordering is especially important: in particular, the derivation of an ordering strategy which fits the three-dimensional reduced grid. Such orderings are superior to natural lexicographic orderings or general ordering strategies which do not take the special structure of the reduced grid and the reduced computational molecule into account.

The most important advantage of the cyclically reduced system is that there is typically an improvement in the condition number by a factor of 2 to 3. This directly affects the rate of convergence. As has been shown, for block stationary methods the bounds on convergence rates can be referred to as a reliable indication of the gains, and here the ratio between the bounds on convergence rates of the reduced system and the unreduced system goes up to approximately $\frac{8}{3}$ for the 1D partitioning, and slightly less for the 2D partitioning.

On the other hand, the 3D cyclically reduced system has some disadvantages; the most serious one is: more significant loss of sparsity compared to lower dimensions, which causes an increase in computational work involved in a single matrix-vector product, and in construction of preconditioners. In fact, the three-dimensional problem is different from the 1D and 2D problems, in that it is the only case where the number of nonzeros of the reduced matrix is higher than the number of nonzeros of the unreduced matrix: about 35% more nonzeros in the 3D case, as opposed to nearly 50% and 10% fewer nonzeros in the 1D and 2D cases respectively.

Another disadvantage is that in general, setting up the reduced system is more difficult than using other finite difference schemes, due to the structure of the computational molecule and the structure of the grid.
Chapter 6. Summary, Conclusions and Future Research

Despite these disadvantages, one step of cyclic reduction is highly effective: the significant gain in iteration counts compensates for the more expensive matrix-vector products, and in practice the solution time has been observed to be typically faster by a factor of two or more. The difficulty in implementation is minor; implementation can be done by performing the step of block Gaussian elimination directly, which is easy. Alternatively, the difference operator, which is given explicitly in Chap. 2, can be used.

The above discussion leads to the conclusion that one step of cyclic reduction for three-dimensional problems is an effective preconditioning technique, which leads to faster solution of linear systems associated with three-dimensional elliptic problems.

6.2 Future Research

A few possible future research activities are:

1. Application of the cyclic reduction step to other discretization schemes. The cyclic reduction step is effective for matrices which satisfy Property A. When the PDE has rough or discontinuous coefficients there are more appropriate schemes, in which there is strong coupling between the red and the black points. When considering cyclic reduction for schemes whose associated matrix does not satisfy Property A, one potential difficulty is that the matrix to be inverted is not diagonal. As a result:

   - The step of cyclic reduction requires more construction time and more computational work per iteration.
   - It is difficult to derive a difference operator and perform convergence analysis.

On the other hand, the significant gains in computational work that have been presented in this thesis lead one to believe that cyclic reduction might still be highly effective.

Some numerical experiments that have been performed as part of this research on Roe & Silikover’s optimum positive residual scheme [92] for three-dimensional problems with constant coefficients, demonstrate significant gains in iterations for the reduced system but
considerably larger computational work per iteration. Careful investigation and imple-
mentation are required in order to reach definite conclusions as to whether cyclic reduction
is effective here. As part of application of the cyclic reduction step to other discretization
schemes, higher order schemes and linear systems associated with finite elements are of
much interest.

2. **Extend the class of problems.** Navier-Stokes equations, which are difficult to solve
numerically, translate into a \(2 \times 2\) block indefinite linear system of equations, which
contains “a convection-diffusion matrix”. Here cyclic reduction can be used as an inner
iteration.

3. **Ordering strategies.** The framework of block orderings which has been described for
the reduced grid in Chap. 3 is an effective tool, and research in this area should be
pursued. Questions of interest are whether there can be obtained a ”nearly-optimal”
ordering strategy, based on point and block computational molecules and grids, along
with other considerations.

An interesting idea of Golub [53] is a *dynamic* change of direction of the ordering through-
out the iteration. Such an adaptive ordering strategy does not seem to have been applied,
and might be a promising technique, in particular when the direction of the flow is not
clear.

4. **Parallelism.** As already mentioned, parallelism is important in particular for three-
dimensional problems, due to the size of the discretized systems. The cyclically reduced
operator has some features that are different from standard finite difference operators. In
particular, the structure of the computational molecule is more complicated. A possible
idea is to map unknowns into processors not necessarily in the standard way; rather,
assign subcubes whose sides form a \(45^\circ\) angle in all of \(x, y\) and \(z\) directions, relative to
the axes. This fits the structure of the reduced computational molecule, and might reduce
the communication time required between processors in the network. On the other hand,
it requires a nontrivial treatment of the grid, for example close to the boundaries.
Implementation in a parallel environment requires further investigation.

5. Analysis of the cyclically reduced operator as a smoother for multigrid algorithms. Preliminary smoothing analysis suggests that the cyclically reduced operator is a good smoother for multigrid, and generates smaller smoothing factors than the analogous seven-point operator. In multigrid, since it involves several grids, and since on coarser grids the mesh Reynolds numbers grow larger, the fact that the cyclically reduced difference equation corresponds to adding artificial viscosity to the original equation suggests that in convection-dominated PDEs the gain of applying cyclic reduction might be significant. An indication for this is the stability of the block Gauss-Seidel scheme, when applied to the cyclically reduced linear system in convection-dominated regions. Research on this topic is currently underway.
Bibliography

perturbed parabolic problems. Technical report, Scientific Computation Research Center,


[3] P. Amadino and M. Paprzycki. A cyclic reduction approach to the numerical solution of

1956.

[5] W. E. Arnoldi. The principle of minimized iteration in the solution of the matrix eigen-

[6] U. M. Ascher. Introduction to multigrid. lecture notes, University of British Columbia,
1997.


[11] Richard Barrett, Michael Berry, Tony F. Chan, James Demmel, June Donato, Jack Dong-
arra, Victor Eijkhout, Roldan Pozo, Charles Romine, and Henk van der Vorst. Templates for
the Solution of Linear Systems: Building Blocks for Iterative Methods. SIAM, Philadelphia,
1994.

[12] I. Bauer and L. Reiss. Block five diagonal matrices and the fast numerical solution of


[14] M. Benzi, D. B. Szyld, and A. van Duin. Orderings for incomplete factorization precondi-
tioning of nonsymmetric problems. Technical Report 97-91, Temple University, Philadel-
phia, 1997.


Bibliography


Bibliography


[53] G. H. Golub. personal communication.


163
Bibliography


Bibliography


