DISCRETIZATION AND SOLUTION OF ELLIPTIC PDEs : A TRANSFORM DOMAIN APPROACH

by

Chung-Chieh Kuo

B.S., National Taiwan University (1980)
S.M., Massachusetts Institute of Technology (1985)
E.E., Massachusetts Institute of Technology (1985)

SUBMITTED TO THE DEPARTMENT OF
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
August, 1987

Signature of Author: [illegible]
Department of Electrical Engineering and Computer Science
August 7, 1987

Certified by: [illegible]
Associate Prof. Bernard C. Levy, Thesis Supervisor

Certified by: [illegible]
Associate Prof. John N. Tsitsiklis, Thesis Supervisor

Accepted by:
Prof. Arthur C. Smith,
Chairman, Departmental Committee on Graduate Students

Discretization and Solution of Elliptic PDEs : A Transform Domain Approach

by

Chung-Chieh Kuo

Submitted to the Department of Electrical Engineering and Computer Science on August 7, 1987 in partial fulfillment of the requirements of the degree of Doctor of Philosophy

Abstract

This thesis seeks to combine numerical analysis, digital filtering techniques, and parallel computer architectures to achieve highly parallel computational schemes for solving elliptic partial differential equations (PDEs). Four main topics are considered : finite-difference discretization formulas, single-grid solution methods, multigrid solution methods and parallel processing. In the area of finite-difference discretization formulas, a new mode-dependent finite-difference method is developed. In the area of single-grid methods, two algorithms, namely, a local relaxation method and a two-level four-color SOR method are presented. In the context of multigrid methods, multigrid algorithms employing the red/black Gauss-Seidel and SOR smoothers are studied. Finally, in the area of parallel processing, the implementation of single-grid and multigrid solution methods on multiprocessor arrays is examined. Throughout this thesis, a unified transform domain approach is used to analyze and design the discretization and solution procedures which are developed.

These Supervisor:	Bernard C. Levy
Title:	Associate Professor of Electrical Engineering
These Supervisor:	John N. Tsitsiklis
Title:	Associate Professor of Electrical Engineering

Acknowledgements

I would like to thank my thesis supervisors, Professors Bernard Levy and John Tsitsiklis for their help and encouragements. I particularly appreciate the freedom they gave me to pursue various interesting research topics. My work with them has been a truly enjoyable experience. Special thanks are due to Professor Levy. Since I came to MIT, he has been taking care of me in many aspects. His care and guidance over the past five years is believed to be one of the best blessings in my life. I would like to express my deep gratitude to Professor Nick Trefethen. He lead me into the rich world of numerical analysis and on numerous occasions provided me with valuable advice concerning my research, writing and career decision. I am also grateful to Professor Bruce Musicus for his kindness and many suggestions, and to Dr. Shlomo Ta'asan for helpful discussions on multigrid methods.

I wish to acknowledge the financial support for this research which has been provided in part by the Army Research Office under grant numbers DAAG29-84-K-0005 and DAAL03-86-K-0171, and in part by the Advanced Research Projects Agency monitored by ONR under contract N00014-81-K-0742, and by AFOSR Contract F49620-84-C-0004.

During my stay at MIT, I have had the opportunity to interact closely with several members of the MIT faculty. Among these, I would like to thank particularly Professors Sanjoy Mitter, Dimitri Bertsekas, Robert Kennedy, Jin-Au Kong, and Campbell Searle. Throughout my graduate years, I have worked at the Laboratory of Information and Decision Systems (LIDS) and have also been closely associated with the Digital Signal Processing Group (DSPG) of the Research Laboratory of Electronics (RLE). I am very grateful to many friends in these two laboratories for their care, sharing and friendship, which greatly enriched my years at MIT. In particular, I would like to thank Kevin Tsai, Whay Lee, Jean R'egnier, John Spinelli, Ellen Hahne, Patrick Hosein, Utpal Mukherji, Ying Li, Rajesh Pankaj, Emre Telatar, Zhi-Quan Luo, Chonghwan Lee, Chee-Seng Chow, George Hart, Thomas Richardson, Ali Ozbek, Ahmed Tewfik, Ken Chou, Xi-Cheng Lou, Soung Liew, Albert Wong, Jerome Abernathy, Louise Alterman and Betty Lou in LIDS and Hong Jeong, Avideh Zakhor, Thrasyvoulos Pappas, Prasanna, Rosalind Wright, Daniel Cobra, Meir Feder, Michele Covell, Giovanni Aliberti, Cory Myers, Patrick Van-Hove, Web Dove, Evangelos Milios, Dennis Martinez and Deborah Gage in RLE.

I owe my deep appreciation to my brothers and sisters of the Church in Cambridge. Praise the Lord, through them, I got saved. Their prayers and unfailing love strengthened and supported me at various stages of my graduate study. Finally, I would like to thank my family, especially my father and grandmother, to whom I dedicate this thesis.

Blessed be the name of God for ever and ever,
to whom belong the wisdom and might.
He changes times and seasons;
he removes kings and sets up kings;
he gives wisdom to the wise
and knowledge to those who have understanding;
he reveals deep and mysterious things,
he knows what is in the darkness,
and the light dwells with him.
To thee, O God of my fathers,
I give my thanks and praises,
for thou hast given me wisdom and strength,
and hast now made known to me what we asked of thee,
for thou hast made known to us the king's matter.

Daniel 2 : 20-23

Contents

List of Figures

List of Tables

Chapter 1 : Introduction

1.1 Motivation

Research on parallel computer architectures and parallel algorithms for solving partial differential equations (PDEs) has progressed very rapidly during the last 15 years [43]. The emergence of this field as an important area of scientific research is motivated by the desire to solve *large scale* scientific and engineering problems such as those arising in weather forecasting, aerodynamic simulation and geophysical exploration, as well as *real-time* problems of the type arising in military applications, robotic control and process control, among many other examples. In order to meet these demanding computational requirements, parallel processing is indispensable [15]. The objective of this thesis is to analyze and design highly parallel computation schemes for the solution of elliptic PDEs. Various aspects of the numerical solution of elliptic PDEs are investigated. The aspects that we consider include the development of finite-difference discretization formulas, both single-grid and multigrid solution methods and their parallel implementation on multiprocessor arrays.

The research described in this thesis has two important features. First, in contrast with the conventional space domain approach, where the solution of elliptic PDEs is based on a sparse matrix formulation, a transform domain approach is used throughout. This approach is motivated by the digital filtering techniques used in the field of digital signal processing (DSP). From this point of view, each local operator is treated as a digital filter and is studied according to its spectral properties. The transform domain approach not only provides a convenient analytical tool, but also puts in evidence certain design issues. As a result, it provides new insights into the choice of discretization and solution schemes for elliptic PDEs, so that we are able to develop new versatile and efficient techniques.

Second, the interaction between computational algorithms and their supporting computer architectures is emphasized. One approach to parallel computation is primarily *architecture*-oriented [30] [33]. The goal of this approach is to build general purpose supercomputers which use parallelism to get better performance. Current supercomputers such as the Cray X-MP and the Cyber 205 use pipelining and vector processing to achieve a high throughput and, generally speaking, these computers are presently the most powerful computing machines for solving PDEs. In spite of their great power, the use of supercomputers remains limited because of cost considerations. A more economical way to achieve a high computing capability consists of using an array processor such as the FPS AP-120B attached to a minicomputer, say a VAX-11, where the attached processor is mainly used for number crunching purposes, and pipelining is used within this processor to speed up computations. Another approach, which is *algorithm*-oriented, has received a large amount of attention recently [12]. This point of view is stimulated by VLSI technology, which makes it economically feasible to build versatile special purpose architectures. Earlier work on systolic and wavefront arrays has primarily focused on matrix computation and signal processing algorithms [35]-[37]. This thesis adopts an algorithm-oriented approach whose objective is to take full advantage of the special structure of PDE problems. Specifically, the locality property of discretized PDEs is exploited to obtain parallel algorithms for their solution and, hence, these algorithms can be conveniently implemented on special purpose processor arrays whose architectures are selected to match the algorithms.

The work of this thesis in fact belongs to an interdisciplinary research area that lies at the intersection of three fields: numerical PDE algorithms, digital filtering techniques and parallel computer architectures (see Figure 1.1). The relationship between these three areas will be emphasized in the introductory chapter. Specifically, the link existing between the theory of digital filters and the numerical solution of PDEs will be explained in Section 1.2, and an overview of current developments in parallel architectures and algorithms for solving numerical PDEs will be presented in Section 1.3. The contributions and organization of this thesis are then outlined in Sections 1.4 and 1.5.

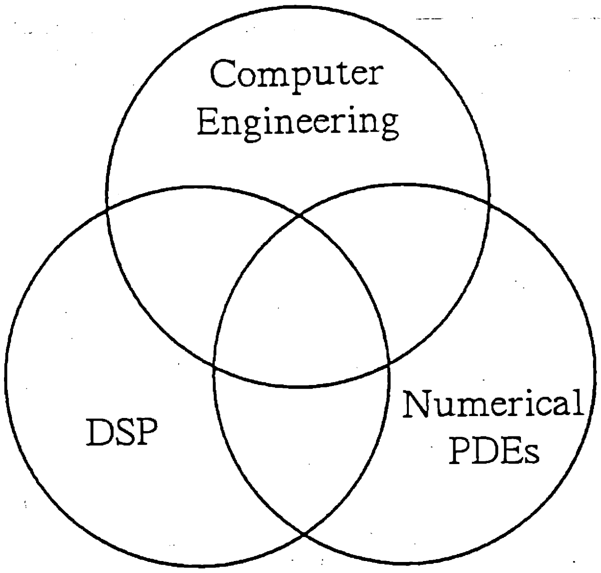


Figure 1.1: Interdisciplinary research area among three fields

1.2 Numerical Solution of PDEs and Digital Filtering

Digital filters are generally defined as a way to process a uniformly sampled data sequence defined on a one-dimensional or multidimensional domain. Usually, we are interested in linear digital filters. For example, consider an input sequence u_n and an output sequence v_n which are related by the following formula

$$\sum_{l=-\infty}^{\infty} a_l u_{n-l} = \sum_{l=-\infty}^{\infty} b_l v_{n-l} , \qquad n \in I . \tag{1}$$

The specification of the coefficients a_l and b_l defines a 1D linear filter. This definition can easily be generalized to higher dimensional spaces [21].

There is a close relationship between the analysis and design of linear digital filters and the transform domain approach. If the coefficients a_l and b_l in (1) are independent of the index n of the sequences u_n and v_n, (1) is a linear shift-invariant (LSI) system whose eigenfunctions are given by e^{iknh}, where h is the distance between two consecutive sampling points and k is arbitrary. Therefore, transform methods such as the discrete-time Fourier transform and the Z-transform provide a natural way to analyze and synthesize LSI systems [42].

For the numerical solution of elliptic PDEs, linear digital filters arise in many different situations. For example, the finite-difference methods which are used to discretize PDE operators can be interpreted in terms of digital filters, as well as the iteration (or smoothing) operators required to solve elliptic PDEs iteratively, or even the interpolation and restriction operations associated to the transfer of data from one grid to another in multigrid methods. The problem that concerns us in this context is that of selecting a performance criterion for each filter and finding some coefficients a_l

and b_l so that the resulting filter has the best possible performance. The first issue corresponds to the problem of *filter specification* and the second one corresponds to that of *filter design*.

The transform domain approach associated with the digital filtering concept can be rigorously applied to simple model problems such as constant-coefficient PDEs with Dirichlet or periodic boundary conditions on a rectangular domain. Hence, for most PDE problems of interest, the transform domain approach is usually regarded as heuristic. In spite of this shortcoming, it is one of the most powerful tools for understanding the numerical behavior of PDE algorithms. Efforts to make transform domain analysis rigorous have been made recently [7][28][29], and further developments in this direction are expected. Nevertheless, a rigorous treatment of the transform domain analysis for general PDEs is not a major concern in this thesis. Instead, we are primarily interested in using the transform domain approach to develop new discretization and solution schemes. These schemes are derived by using a local analysis, where we assume that the local differential operator is linear and has constant coefficients. The analytic results that we obtain are therefore valid for the case where the global operator happens to be a linear constant-coefficient operator. For general PDEs with spatially-varying coefficients, the efficiency of these schemes will be tested by computer simulations.

The main goal of this section is to describe more precisely the link existing between the numerical solution of PDEs and digital filtering. We consider a simple model problem consisting of the 1D Poisson equation

$$\frac{d^2u_c(x)}{dx^2} = f_c(x), \qquad x \in \Omega = [0,1] \tag{2}$$

where $u_c(0)$ and $u_c(1)$ are given. In this section, the subscript c is used to distinguish continuous functions and their Laplace transforms from sampled functions and their Z-transforms.

1.2.1 Digital Filters for Finite-difference Discretization

A general finite-difference discretization scheme for the 1D Poisson equation assumes the form

$$\sum_{l=-L_1}^{L_1} a_l u_{n-l} = \sum_{l=-L_2}^{L_2} b_l f_{n-l}, \tag{3}$$

where u_n is the estimate of the value $u_c(nh)$ and $f_n = f_c(nh)$ is the sampled driving function. Equation (3) is known among numerical analysts as the HODIE (High-Order Difference approximation with Identity Expansions) or OCI (Operator Compact Implicit) scheme [4][39]. From a digital filtering viewpoint, (3) defines an infinite impulse response (IIR) filter [42]. We want to find coefficients a_l and b_l so that (2) is well approximated by (3).

Let us view $u_c(x)$ and u_n as inputs and $f_c(x)$ and f_n as outputs. In the interior region of Ω, we can ignore the effect of boundary conditions and treat (2) and (3) as if they were valid over an infinite domain. Then, by using the Laplace and Z transforms, (2) and (3) can be represented in the transform domain by two block diagrams as shown in Figure 1.2(a) and (b), where $\hat{H}_c(s)$ and $\hat{H}(z)$ are the system functions of (2) and (3) [42] and $\hat{u}_c(s)$, $\hat{u}(z)$, $\hat{f}_c(s)$ and $\hat{f}(z)$ are the transforms of $u_c(x)$, u_n, $f_c(x)$ and f_n, respectively.

The system function $\hat{H}_c(s)$ is the spectrum of the ideal filter that we want to approximate and $\hat{H}(z)$ is the spectrum of the discrete filter under construction. The determination of the coefficients a_l and b_l depends on what criterion we adopt to specify how $\hat{H}(z)$ and $\hat{H}_c(s)$ must resemble each other. Hence, in this framework, the finite-difference discretization problem can be viewed as a filter specification problem.

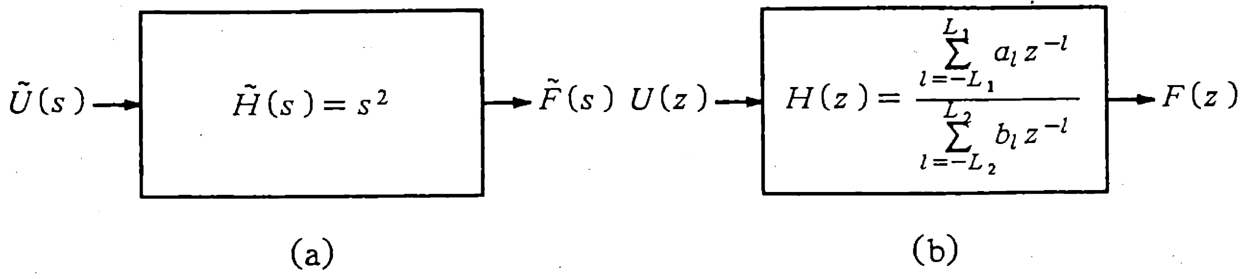

$\tilde{U}(s)$
$\tilde{H}(s)=s^2$
$\tilde{F}(s)$
(a)
$U(z)$
$H(z)=\frac{\sum_{l=-L_1}^{L_1} a_l z^{-l}}{\sum_{l=-L_2}^{L_2} b_l z^{-l}}$
$F(z)$
(b)

Figure 1.2 : Digital filter for the finite-difference discretization

The specification of various 1D and 2D filters for approximating the ordinary and partial differential operators will be discussed in Chapter 2.

1.2.2 Digital Filters for Relaxation

Suppose that (2) is discretized by a 3-point central difference scheme as

$$u_{n-1} - 2u_n + u_{n+1} = h^2 f_n \ .$$

The damped Jacobi relaxation scheme and the SOR scheme with red/black partitioning are two relaxation methods which are commonly used for solving this equation. They are both parameterized by a relaxation parameter ω and are defined respectively as

$$u_n^{m+1} = (1-\omega)u_n^m + \frac{\omega}{2}\left(u_{n-1}^m + u_{n+1}^m - h^2 f_n \right),$$

and

$$u_n^{m+1} = (1-\omega)u_n^m + \frac{\omega}{2}\left(u_{n-1}^m + u_{n+1}^m - h^2 f_n \right), \qquad n \text{ even},$$

$$u_n^{m+1} = (1-\omega)u_n^m + \frac{\omega}{2}\left(u_{n-1}^{m+1} + u_{n+1}^{m+1} - h^2 f_n \right), \qquad n \text{ odd}.$$

Define the error at the m th iteration as $e_n^m = u_n^m - \bar{u}_n$, where $\bar{u}_n$ denotes the exact solution and where u_n^m is the approximation of u_n obtained at the m th iteration. The error dynamics for the damped Jacobi relaxation can be written as

$$e_n^{m+1} = (1-\omega)e_n^m + \frac{\omega}{2}\left(e_{n-1}^m + e_{n+1}^m \right). \tag{4}$$

By treating the error at the m th iteration as the input and the error at the $m+1$th iteration as the output of a digital filter, we find that (4) corresponds to a finite-impulse response (FIR) filter [42]. The frequency response of (4) is represented by the block diagram of Figure 1.3.

$$\begin{tikzcd} E_k^m \arrow[r] & \boxed{(1-\omega) + \omega\cos(kh)} \arrow[r] & E_k^{m+1} \end{tikzcd}$$

Figure 1.3 : Digital filter for the damped Jacobi iteration

The equation describing the error dynamics for the red/black SOR scheme is given by

$$e_n^{m+1} = (1-\omega)e_n^m + \frac{\omega}{2}\left(e_{n-1}^m + e_{n+1}^m \right), \qquad n \text{ even}, \tag{5a}$$

$$e_n^{m+1} = (1-\omega)e_n^m + \frac{\omega}{2}\left(e_{n-1}^{m+1} + e_{n+1}^{m+1} \right), \qquad n \text{ odd}. \tag{5b}$$

The above system of equations is not shift-invariant. To transform it into a shift-invariant system, let us construct two sequences from e_n: the red sequence r_n and the black sequence b_n. The red sequence is defined as

$$r_n^m = e_n^m , \qquad n \quad \text{even} ,$$

while r_n^m with odd index n is obtained by interpolating the value of r_n^m with even index. Similarly, the black sequence is defined as

$$b_n^m = e_n^m , \qquad n \ \text{odd} ,$$

and b_n^m with even index n is obtained by interpolating b_n^m with odd index (see the discussion in Chapter 4). Then, (5) can be written as an LSI system where

$$r_n^{m+1} = (1-\omega) r_n^m + \frac{\omega}{2} (b_{n-1}^m + b_{n+1}^m) , \quad \text{for all } n , \tag{6a}$$

$$b_n^{m+1} = (1-\omega) b_n^m + \frac{\omega}{2} [(1-\omega)(r_{n-1}^m + r_{n+1}^m) + \frac{\omega}{2}(b_{n-2}^m + 2b_n^m + b_{n+2}^m)] , \quad \text{for all } n . \tag{6b}$$

Therefore, a transform domain analysis can be applied to (6). By using the discrete Fourier transform, (6) can be represented by the diagram shown in Figure 1.4.

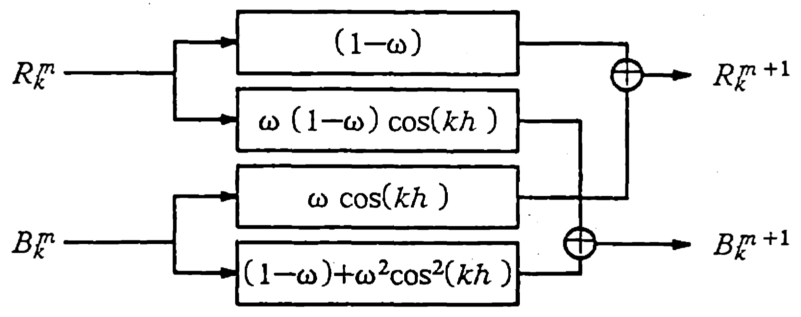


Figure 1.4: Digital filter for the red/black SOR iteration

In terms of filter specification, the relaxation filter has to satisfy two requirements. First, the filter corresponding to an iterative algorithm has to be consistent with the given finite-difference equation. If the input sequence of the filter is the solution $\bar{u}_n$ of the system of difference equations, the output should also be the same sequence $\bar{u}_n$. Second, since this filter can be interpreted as a smoothing filter which reduces the magnitude of all the Fourier components of the initial error in the transform domain, the magnitude of the frequency response of the filter for all feasible error components should be less than one so that all error components will be reduced to zero eventually.

There are many different filters satisfying the above two requirements. Thus we are lead to a filter design problem. Note that the filters for the damped Jacobi and the SOR iterations have very different structures. In addition, for each type of filter, the appropriate relaxation parameter ω can be determined according to different criteria.

Digital filters used to smooth out iteratively the error for ODEs and PDEs are discussed in Chapters 3, 4 and 5.

1.2.3 Digital Filters for Restriction and Interpolation

Another type of digital filter appears when functions are transferred between two grid levels in multigrid algorithms [5]. Consider two grids Ω_h and $\Omega_{h'}$ with $h' = 2h$. A restriction operator $R_h^{h'}$ which maps a function u_i defined on Ω_h into a function $w_{i'}$ defined on $\Omega_{h'}$ can be defined as follows

$$w_{n'} = \sum_{l=-L}^{L} a_l u_{2n'-l} \; .$$

Similarly, an interpolation operator $I_{h'}^{h}$ which maps a function $w_{i'}$ defined on $\Omega_{h'}$ into a function u_i defined on Ω_h can be defined as

$$u_{2n'} = w_{n'} \quad \text{and} \quad u_{2n'+1} = \sum_{\substack{l=-L \\ l \text{ odd}}}^{L} b_l w_{n'+\frac{1}{2}-\frac{l}{2}} \; .$$

Conceptually, it is convenient to view the restriction operator $R_h^{h'}$ as composed of two steps [18]:

Step 1: low-pass filter on the fine grid Ω_h ,

$$v_n = \sum_{l=-L}^{L} a_l u_{n-l} \quad ,$$

where v_i is also defined on Ω_h , and

Step 2: down-sampler from the fine grid Ω_h *to the coarse grid* $\Omega_{h'}$,

$$w_{n'} = v_{2n'} .$$

Similarly, the interpolation operator $I_{h'}^{h}$ can be conceptually decomposed into two steps:

Step 1: up-sampler from the coarse grid $\Omega_{h'}$ *to the fine grid* Ω_h ,

$$v_{2n'} = w_{n'} , \quad v_{2n'+1} = 0 .$$

Step 2: low-pass filter on the fine grid Ω_h ,

$$u_n = \sum_{l=-L}^{L} b_l v_{n-l} .$$

Since restriction and interpolation operators are lowpass filters cascaded with simple up-sampling and down-sampling operations, the above decompositions have the effect of reducing the design of restriction and interpolation operators to that of appropriate low-pass filters. Block diagrams illustrating the above two decompositions are presented in Figure 1.5. The design of restriction and interpolation schemes based on these ideas will be treated in Sections 5.3 of Chapter 5.

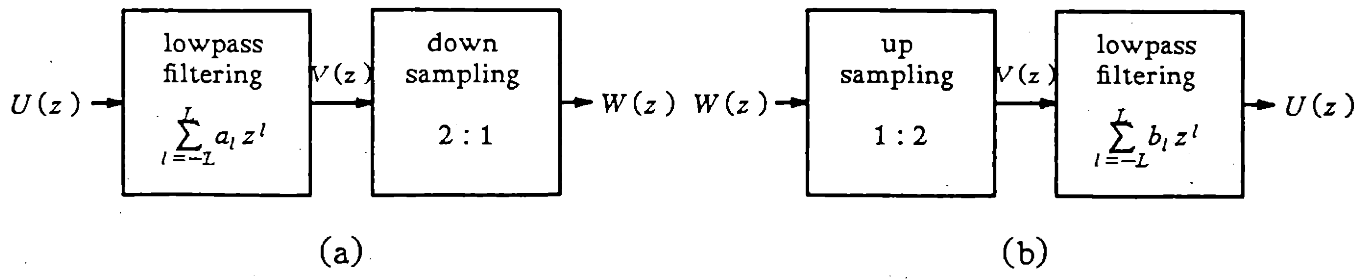


Figure 1.5 : Digital filters for the restriction and interpolation

1.3 Parallel Computation of PDEs : An Overview

This section contains a brief review of previous work on parallel computation of PDEs. First we look at two architectures which are currently available for large scale computation, namely, vector computers and attached array processors. Then we examine new architectures either in the experimental stage or still under investigation, and discuss some issues related with the implementation of parallel PDE algorithms.

We do not intend to be exhaustive in this brief review, since a survey has been published recently by Ortega and Voigt [43], and there are also several textbooks devoted to parallel computers and parallel computation such as [32] and [33]. This section attempts only to present an overview of current research activities centered around parallel computation of PDEs and, in the process we point out the important issues related to each approach.

1.3.1 Current Technology for Large-Scale Computation

(1) Vector Computers

Vector computers such as the Cray 1, Cyber 205, and Fujitsu VP-200 are the most powerful computing machines currently available for scientific and engineering applications, and the peak computation speed which can be achieved by these computers is in the range of 100 - 500 megaflops (million floating-point operations per second). The architectures of the Cray 1 and Cyber 205 were described in [46] and [38] respectively, and the architecture of the Fujitsu VP-200 computer was discussed in a special issue of IEEE Computer magazine [17]. New vector computers such as the Cray X-MP and the Cray 2 which connect several vector machines together have also come to the market recently. The maximum computation speed for these machines is expected to be 1 - 10 gigaflops (billion floating-point operations per second). The application of vector computers to the solution of PDEs is discussed in Rodrigue [45] and in a special issue of the Proceedings of IEEE [44], among other conference proceedings and reports.

The two key features on which all vector computers rely to obtain high performance computation are special hardware and firmware design suitable for vector operations, and highly pipelined systems including instruction pipelining and arithmetic pipelining. As a consequence, the efficiency of these machines depends on two factors: (1) how to achieve the maximum pipelining [34], and (2) how to get the maximum amount of vector operations. The percentage of code that can be vectorized ranges from 10 to 90 percent for a broad range of problems and the remaining non-vectorizable scalar operations constitute the major bottleneck of these vector machines.

(2) Attached Array Processors

Another popular approach to obtain high computation speeds is to use an array processor attached to a host computer. This combination can achieve a computation speed of 10 megaflops; however, it costs much less than vector supercomputers. There have been two special issues on attached array processor architectures and applications in Computer magazine [11] [14]. The use of an attached array processor to solve elliptic PDEs has also been discussed by Schultz [48].

One problem with this approach is that programming array processors usually takes more time. Furthermore, this approach is only appropriate for problems of moderate size [14].

1.3.2 Parallel Computer Architectures

One simple way to construct a large parallel computer is to connect several vector computers together. Architectures of this type have been tested at the Lawrence Livermore National Laboratory and designed by some companies, but it is not clear that this architecture is the right one to use for parallel computation in the long run. Researchers are currently developing some alternate architectures among which we will mention data flow computers, multiprocessor systems (MIMD machines), and processor arrays (SIMD machines).

(1) Data Flow Computers

Data flow computers are based on the data driven computation concept. Research on data flow computers was initiated with the analysis of data flow program graphs and was influenced by the development of functional languages. Most of the early work in this area has focused on the design of data flow languages and their compilers. A few prototype machines are presently being built. The structure of a practical data flow machine was described in [27].

Among various data flow computer projects, we can list the Dennis static data flow machine and the Arvind dynamic data flow machine at MIT, the DDM project at the University of Utah, the EDDY system in Japan, and the Manchester machine in England. A special issue on data flow computers appeared in the IEEE Computer magazine [13], which presented an overview of research up to that time.

The advantages of data flow computers include the potential to achieve high parallelism at the instruction level, the ability to issue multiple memory requests, and the functional programming style that allows easier proofs of correctness. However, there are also some disadvantages, among which are a high system overhead to detect parallelism and to schedule the available processors, difficulties in treating complex data structures, and the lack of memory hierarchy [3] and [23]. Another problem is with the hardware implementation, namely, the design of the switching network. This problem seems solvable, but, it will limit the performance of such machines if the size of the network becomes too large.

PDE problems are an important application area for data flow computers and the implementation of PDE solvers has been the subject of some studies [2] [20].

Although data driven computing is an interesting and attractive idea, many problems still need to be solved. Other researchers are looking at approaches where a number of traditional von Neumann type machines are connected together to obtain a parallel machine with improved computation speed. The machines which are obtained by this process can be divided into two categories: MIMD (Multiple Instruction Multiple Data) machines and SIMD (Single Instruction Multiple Data) machines.

(2) Multiprocessor Systems (MIMD Machines)

An MIMD machine can be viewed as a network of interconnected minicomputers, each of which has its own memory, control commands, and even I/O devices, and can operate asynchronously and independently of other processors. However, some of these processors may also share common resources, say a common memory and I/O channels. In some sense, an MIMD machine is similar to a local area network system. However, the goal in designing an MIMD computer is to obtain a general purpose high speed computing machine, while the design of a local area network merely aims at resource sharing and information exchange. Since the goals are different, the design considerations will also be different. Currently, MIMD projects are trying to connect tens or hundreds of processors together. In the future, it is expected that thousands of processors will be connected into a "big" parallel machine. The performance of each processor may be similar to or better than that of a minicomputer such as the VAX family. Assuming that each processor can perform 10 megaflops and that all processors are busy solving a single problem, the maximum speed that can be achieved with one thousand processors and a "smart" communication scheme among them is 10 gigaflops.

There are many MIMD machine projects in universities, such as the Ultracomputer at N.Y.U., the TRAC (Texas Reconfigurable Array Computer) project at the University of Texas, the Cedar project at the University of Illinois, the Dynamic Multicomputer System at the University of Florida, the Database Machine at the University of Wisconsin, the PASM project at Purdue University, and the Cosmic Cube project at Caltech. These projects are funded by DOE, DARPA, NSF, ONR, and NASA [47]. Some companies in industry are also building commercial multiprocessor systems such as hypercube computers, tree machines, and so on. The MIMD machines are usually designed as general purpose computers. However, some MIMD machines are designed for specific applications, such as the FEM (Finite Element Machine) at the NASA/Langley Research Center for structural engineering applications, and the NSC (Navier-Stokes Computer) at Princeton University for solving fluid dynamic problems.

The key to the success of MIMD machines is the communication scheme among the processors inside each machine [50] [52]. Other important issues are the partitioning of problems, the mapping of the partitioned problems to processors [25], and an efficient access to shared resources [24]

Some preliminary studies on the use of hypercube computers to solve PDEs have been performed in [9] and [10].

(3) **Processor Arrays (SIMD Machines)**

An SIMD machine is also an interconnected network of multiple processors. However, these processors are much simpler, have each a small local memory and an ALU, communicate only with the host computer or neighboring processors, and usually operate under a synchronized global clock. Typical examples are the Illiac IV, systolic arrays, the MPP (Massively Parallel Processor), and the Connection Machine. Current SIMD machines under development include the Warp machine (a linear 10 processor systolic array) at CMU, and the blue chip project at Purdue.

It is feasible nowadays to connect hundreds of processors together, and it may be possible to obtain an array with 10^4 processors or more before long. Let us roughly estimate the computation capability of such an array. Assuming that the maximum speed of each processor is 10 megaflops and that all processors are busy in solving the same problem with local communication only, then the maximum computation speed of such an array is 100 gigaflops or more, which is about 1000 times faster than the Cray 1.

The success of SIMD machines depends on how algorithms and architectures are matched. If there exists a highly parallel algorithm with a well matched architecture for some important applications, SIMD machines may provide the most advantageous solution. Since parallelism is exploited at the algorithmic level, the overhead in detecting parallelism is not necessary and the control can be much simpler.

1.3.3 Parallel Numerical PDE Algorithms

So far, there are few algorithms which were truly developed from a parallel computation point of view. Most parallel algorithms can be viewed as *parallel implementations* of traditional algorithms. As a consequence, we cannot separate parallel algorithms from parallel computers. For different parallel machines, the criterion of parallelizability can be quite different. For example, for a vector computer, parallelization is more or less the same as vectorization and communication constraints are not important. However, communication requirements become critical for multiprocessor systems.

An ideal parallel computer model, called PRAM (Parallel Random Access Machine), is sometimes used to illustrate parallel algorithms [49]. In this model, we assume that an infinite number of processors are connected to a shared central memory and that each read or write operation on a memory cell for any processor takes one unit cycle. Algorithms based on this model may be of theoretical interest; however, this machine model is not realistic. The implementation of PRAM by a real machine usually needs an overhead of $O(\log N)$ to achieve the communication requirements.

Some researchers have proposed parallel algorithms to solve a linear system of equations which use more than $O(N)$ processors, where N is the number of unknowns [35]. These methods can be used for a linear system of moderate size, say, when $N < 100$. However, they are not practical for systems of equations obtained from discretized PDEs, where N is usually larger than 10^4. Therefore, based on this consideration, a realistic multiprocessor system for PDEs should contain at most $O(N)$ processors.

Parallel numerical algorithms have been reviewed by Miranker [40], Heller [31], and Ortega and Voigt [43]. The topics covered in Miranker's paper are quite broad, including optimization, root finding, differential equations, and linear equations. Heller's paper focused on parallel numerical linear algebra, while Ortega and Voigt's paper concentrates on parallel numerical PDE algorithms, which is most relevant to our discussion here. Parallel *direct* methods for solving PDEs may be categorized into three classes: factorization methods - parallel implementations of the Gaussian elimination and Givens rotation methods, ordering methods - parallel implementations of the nested disection method, and special fast algorithms - special methods for tridiagonal systems and fast Poisson solvers. Parallel *iterative* methods can be divided into two classes: parallel implementations of gradient-type algorithms and parallel implementations of relaxation-type algorithms, where in this category we include multigrid methods [8][26]. The details can be found in Ortega and Voigt's paper [43] and the references therein.

1.3.4 Future Trends

In the future, it is expected that both general purpose and special purpose computers will exist, since different users have different needs. In the general purpose computer market, today's vector supercomputers may be replaced by MIMD supercomputers or data flow supercomputers. Whether the processor in an MIMD machine is a vector computer or a minicomputer will be the designers' choice. As to the special purpose computer market, attached array processors may be replaced by processor arrays such as SIMD machines. Since the trend goes from central to distributed systems, researchers in computer architecture will put more effort in the design of interconnection networks. In addition, the development of software for distributed systems such as distributed operating systems, parallel programming languages, and compilers for parallel programs will become more and more important.

Due to the evolution of computer architectures, the research activity in the field of parallel algorithms will shift from "vectorized" parallel algorithms to "distributed" parallel algorithms. In a distributed computational environment, communication complexity in an algorithm is almost as important as computation complexity. A new complexity theory for parallel algorithms should include both. In addition to the parallel implementations of traditional algorithms, we also expect that some new algorithms will be developed which can fully utilize parallel distributed computation structures. However, this kind of research requires a deeper understanding of algorithms and some knowledge of feasible computer architectures [1][6][22].

1.4 Thesis Contributions

The contributions of this thesis are of two sorts: a methodology and a succession of algorithmic results based on it.

The methodology is to apply digital filtering techniques to the analysis and design of numerical PDE algorithms. It appears that this application has not yet been fully pursued so far. The reason for this relative void is that researchers in digital filtering and numerical PDEs are trained in different departments and seldom communicate with each other. As to algorithmic results, several methods concerning the discretization and solution of elliptic PDEs have been developed. These results are summarized below.

Chapter 2: Mode-dependent Finite-difference Method

Finite-difference approximations are not usually studied from the transform domain point of view. Based on this viewpoint, we derive new high order finite-difference schemes. For an Rth order ODE, we develop a systematic approach to obtain an $(R+1)$-point mode-dependent finite-difference scheme, which is exact when used to discretize linear constant-coefficient ODEs. For the two-dimensional second order constant-coefficient Helmholtz and convection-diffusion equations, we obtain 5-point, rotated 5-point, and 9-point stencil discretizations which have accuracy of $O(h^2)$, $O(h^2)$ and $O(h^6)$ respectively.

Chapter 3: Local Relaxation Method

We prove the convergence of a local relaxation method which solves $Ax = b$, where A is symmetric positive definite, by using space-adaptive relaxation parameters.

The local Fourier analysis approach is used to analyze the SOR method with red/black ordering. For the model Poisson problem, the optimal relaxation parameter and the convergence rate of the method obtained are consistent with Young's results [53]. For spatially varying coefficient PDEs, computer simulation shows that the local relaxation method is slightly better than the standard SOR method which uses time-adaptive relaxation parameters. We also derive a local relaxation method for a 9-point stencil discretization of the operator $\frac{\partial^2}{\partial x^2} + a \frac{\partial^2}{\partial x \partial y} + \frac{\partial^2}{\partial y^2}$ and prove its convergence. The convergence rates of the 5-point and the 9-point local relaxation methods are both proportional to $O(h)$.

Chapter 4: Two-level Four-color SOR Method

We study the SOR acceleration effect from a transform domain point of view, and develop a new two-level four-color SOR method for solving the system of equations resulting from the 9-point stencil discretization of the Laplacian. The two-level SOR method contains both block and point iterations, which use different relaxation parameters. A closed form expression is then derived for both the block and point relaxation parameters.

Chapter 5: Two-color Multigrid Methods

We use a variant of Fourier analysis called the *two-color* Fourier analysis to clarify the physical mechanism of a multigrid method which employs the red/black Gauss-Seidel smoothing iteration for a model problem consisting of the Poisson equation on the unit square with Dirichlet boundary conditions.

The two-color Fourier analysis not only serves as an analytical tool but also as a design tool. This is particularly evident for the 1D problem, for which the two-color two-grid Fourier analysis is used to design a fast direct method. For 2D problems, several design issues such as rearranging the smoothing order, and smoothing with a relaxation parameter $\omega \neq 1$ are also investigated from the same viewpoint.

Chapter 6: Parallel Implementations of Single-Grid and Multigrid Methods

We study the communication and computation requirements associated to the implementation of iterative algorithms for elliptic PDEs on a mesh-connected processor array. We show that the acceleration effect of the traditional SOR and conjugate gradient methods is canceled out by the necessity to use global communications at each iteration, and argue that the local relaxation method developed in Chapter 3 is an ideal scheme for parallel implementation on a mesh-connected array. The optimality of this implementation, which requires $O(\sqrt{N})$ total running time, is explained.

We also present a brief survey of current research work on parallel implementations of single-grid and multigrid solution methods on various multiprocessor arrays.

1.5 Organization of the Thesis

This thesis is based on several reports written at different stages of my Ph.D. research. Since each report has its own introduction, conclusion and references, this feature is preserved in each chapter for convenience. An advantage of this organization is that each chapter can be read independently of other chapters. This feature should prove to be convenient for readers who are interested only in a subset of the topics discussed here.

However, efforts have been made to unify all the concepts and notations appearing in the different reports corresponding to each chapter, so that the thesis has its own integrity and consistency.

The link between numerical PDE algorithms and digital filtering theory was discussed above in Section 1.2. Although the topics discussed in this thesis are varied, digital filtering is in fact the main unifying theme behind the various discretization and solution schemes that are proposed. Since Section 1.2 describes the general point of view adopted in this thesis, interested readers will find it helpful to read it first.

Current developments in parallel computer architectures and algorithms for the solution of PDEs were also described in Section 1.3. This survey gives a retrospective of the general field and points out some future research directions. Above all, it provides a justification for the algorithm-oriented approach adopted in this research. Throughout the thesis, we focus on numerical PDE algorithms which can be conveniently implemented on multiprocessor arrays (MIMD or SIMD machines). In such a parallel/distributed computing environment, communication costs have to be taken explicitly into account and, hence, local communication schemes are preferred. Due to this consideration, only finite-difference discretization schemes and iterative algorithms with local operations are examined in this thesis.

A consistent set of notation is adopted in this thesis, which is summarized in Table 1.1.

This thesis consists of four parts: discretization, single-grid and multigrid solution methods, and parallel processing. Each part has it own abstract, so that readers will be able to identify the main issues discussed in each part with a small amount of effort.

Equations are numbered sequentially within each section and referred accordingly in the same section. When they are referred to in other sections, section numbers associated with the desired equations are also included. For example, (1.2.1) denotes equation (1) in Section 1.2.

x, y	spatial domain variables
n, n_x, n_y	integer indices of spatial domain variables
N, N_x, N_y	number of grid points
h, h_x, h_y	grid spacings
$\kappa, \kappa_x, \kappa_y$	wavenumbers
k, k_x, k_y	integer indices of wavenumbers
$\theta, \theta_x, \theta_y$	wavenumber-grid spacing (κh) products
s, s_x, s_y	Laplace transform domain variables
z, z_x, z_y	Z transform domain variables
u_n, v_n, w_n	one-dimensional sequences
$u_{n_x,n_y}, v_{n_x,n_y}, w_{n_x,n_y}$	two-dimensional sequences
$\hat{u}(z)$	Z-transform of u_n
$\hat{u}_k$	Fourier transform of u_n
m	number of iteration
Ω	problem domain
Ω_h	lattice with grid spacing h on domain Ω
Γ	boundary of Ω
Γ_h	boundary of Ω_h
D, D_x, D_y	basic differential operators
E, E_x, E_y	basic shift operators
L	differential operator
L_h	finite-difference operator defined on Ω_h
N_L	nullspace of operator L
C	coincident space
I_h	averaging operator defined on Ω_h
$L_{h,+}$	5-point stencil discretization operator
$L_{h,\times}$	rotated 5-point stencil discretization operator
$L_{h,9}$	9-point stencil discretization operator
$\omega, \omega_b, \omega_p$	relaxation parameters
J	Jacobi relaxation operator
J_ω	damped Jacobi relaxation operator
ρ	spectral radius of J
G	Gauss-Seidel relaxation operator
G_ω	successive over- relaxation operator
G_W	local relaxation operator
λ	spectral radius of G_ω

Table 1.1: Summary of notation

1.6 References

[1] L. Adams and J. M. Ortega, "A Multi-Color SOR Method for Parallel Computation," ICASE report, 82-9, Apr. 1982.

[2] Arvind and R. E. Bryant, "Design Considerations for a Partial Differential Equation Machine," MIT Integrated Circuit Memo No. 80-3, Jan. 1980.

[3] J. L. Baer, "Computer Architecture," *Computer*, vol. 17, no. 10, pp. 77-87, Oct. 1984.

[4] A. E. Berger, J. M. Solomon, M. Ciment, S. H. Leventhal, and B. C. Weinberg, "Generalized OCI Schemes for Boundary Layer Problems," *Math. Comp.*, vol. 35, no. 151, pp. 695-731, Jul. 1980.

[5] A. Brandt, "Multi-level Adaptive Solutions to Boundary-value Problems," *Math. of Comp.*, vol. 31, no. 138, pp. 333-390, Apr. 1977.

[6] A. Brandt, "Multigrid Solvers on Parallel Computers," in *Elliptic Problem Slovers*, ed. M. H. Schultz, New York, N.Y.: Academic Press, Inc., 1981, pp. 39-83.

[7] A. Brandt, "Multigrid Techniques: 1986 Guide," Preprint.

[8] T. F. Chan and R. Schreiber, "Parallel Networks for Multi-Grid Algorithms: Architecture and Complexity," *SIAM J. Sci. Stat. Comput.*, vol. 6, pp. 698-711, Jul. 1985.

[9] T. F. Chan and Y. Saad, "Multigrid Algorithms on Hypercube Multiprocessor," *IEEE Trans. on Computers*, vol. C-35, no. 11, pp. 969-977, Nov. 1986.

[10] T. F. Chan, Y. Saad, and M. H. Schultz, "Solving Elliptic Partial Differential Equations on the Hypercube Multiprocessor," Research Report YALEU/DCS /RR-373, Mar. 1985.

[11] *Computer : Array Processor Architecture*, vol. 14, no. 9, Sep. 1981.

[12] *Computer : Highly Parallel Computing*, vol. 15, no. 1, Jan. 1982.

[13] *Computer : Data Flow Systems*, vol. 15, no. 2, Feb. 1982.

[14] *Computer : Applications for Array Processors*, vol. 16, no. 6, Jun. 1983.

[15] *Computer : The State of Computing*, vol. 17, no. 10, Oct. 1984.

[16] *Computer : Multiprocessing Technology*, vol. 18, no. 6, Jun. 1985.

[17] *Computer : Supercomputer Benchmark*, vol. 18, no. 12, Dec. 1985.

[18] R. E. Crochiere and L. R. Rabiner, *Multirate Digital Signal Processing*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1983.

[19] J. B. Dennis, "High Speed Data Flow Computer Architecture for the Solution of the Navier-Stokes Equations," MIT Computation Structures Group Memo 225, 1982.

[20] J. B. Dennis, G.-R. Gao, and K. W. Todd, "Modelling the Weather with a Data Flow Supercomputer," *IEEE Trans. on Computers*, vol. C-33, no. 7, pp. 592-603, Jul. 1984.

[21] D. E. Dudgeon and R. M. Mersereau, *Multidimensional Digital Signal Processing*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1984.

[22] D. J. Evans, "New Parallel Algorithms for Partial Differential Equations," in *Parallel Computing 83*, ed. U. Schendel, Elsevier Science Publishers (North-Holland), 1984, pp. 3-56.

[23] D. D. Gajski, D. A. Padua, D. J. Kuck, and R. H. Kuhn, "A Second Opinion on Data Flow Machines and Languages," *Computer*, vol. 15, no. 2, pp. 58-69, Feb. 1982.

[24] D. D. Gajski and J.-K. Peir, "Essential Issues in Multiprocessor Systems," *Computer*, vol. 18, no. 6, pp. 9-27, Jun. 1985.

[25] D. Gannon, "On Mapping Non-uniform PDE Structures and Algorithms onto Uniform Array Architectures," in *Proceeding of International Conference on Parallel Processing*, 1981.

[26] D. Gannon and J. Van Rosendale, "Highly Parallel Multigrid Solvers for Elliptic PDEs: An Experimental Analysis," ICASE Report 82-36, 1982.

[27] J. R. Gurd, C. C. Kirkham, and Watson, I., "The Manchester Prototype Dataflow Computer," *Comm. of ACM*, vol. 28, no. 1, pp. 34-52, Jan. 1985.

[28] W. Hackbusch, "Multigrid Convergence for a Singular Perturbation Problem," *Lin. Alg. Appl.*, vol. 58, pp. 125-145, 1984.

[29] W. Hackbusch, *Multi-Grid Methods and Applications*. Berlin, Germany: Springer-Verlag, 1985.

[30] L. S. Haynes, R. L. Lau, D. P. Siewiorek, and D. W. Mizell, "A Survey of Highly Parallel Computing," *Computer*, vol. 15, no. 1, pp. 9-24, Jan. 1982.

[31] D. Heller, "A Survey of Parallel Algorithms in Numerical Linear Algebra," *SIAM Review*, vol. 20, no. 4, pp. 740-777, Oct. 1978.

[32] R. W. Hockney and C. R. Jesshope, *Parallel Computers: Architecture, Programming, and Algorithms*. Bristol, England: Adam Hilger Ltd., 1981.

[33] K. Hwang and F. A. Briggs, *Computer Architecture and Parallel Processing*. New York, N.Y.: McGraw-Hill, Inc., 1984.

[34] P. M. Kogge, *The Architecture of Pipelined Computers*. New York, N.Y.: Hemisphere Publishing Corporation, 1981.

[35] H. T. Kung and C. E. Leiserson, "Systolic Array (for VLSI)," in *Sparse Matrix Proc. 1978*, SIAM, 1979, pp. 256-282.

[36] H. T. Kung, "Why Systolic Architectures?," *Computer*, vol. 15, no. 1, pp. 37-46, Jan. 1982.

[37] S. Y. Kung, "On Supercomputing with Systolic/Wavefront Array Processors," *Proc. of IEEE*, vol. 72, no. 7, pp. 867-884, Jul. 1984.

[38] N. R. Lincoln, "Technology and Design Tradeoffs in the Creation of a Modern Supercomputer," *IEEE Trans. on Computers*, vol. C-31, no. 5, pp. 349-362, May 1982.

[39] R. E. Lynch and J. R. Rice, "A High-Order Difference Method for Differential Equations," *Mathematics of Computation*, vol. 34, no. 150, pp. 333-372, Apr. 1980.

[40] W. L. Miranker, "A Survey of Parallelism in Numerical Analysis," *SIAM Review*, vol. 13, no. 4, pp. 524-547, Oct. 1971.

[41] D. I. Moldovan, "On the Design of Algorithms for VLSI Systolic Arrays," *Proceedings of the IEEE*, vol. 71, no. 1, Jan. 1983.

[42] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1975.

[43] J. M. Ortega and R. G. Voigt, "Solution of Partial Differetial Equations on Vector and Parallel Computers," *SIAM Review*, vol. 27, no. 2, pp. 149-240, Jun. 1985.

[44] *Proc. of IEEE : Special Issue on Supercomputers - Their Impact on Science and Technology*, vol. 72, no. 1, Jan. 1984.

[45] G. Rodrigue, *Parallel Computations*. New York, N.Y.: Academic Press, Inc., 1982.

[46] R. M. Russel, "The Cray-1 Computer System," *Comm. of ACM*, vol. 21, no. 1, pp. 63-72, Jan. 1978.

[47] P. B. Schneck, D. Austin, S. L. Squires, J. Lehmann, D. Mizell, and K. Wallgren, "Parallel Processor Programs in the Federal Government," *Computer*, vol. 18, no. 6, pp. 43-56, Jun. 1985.

[48] M. H. Schultz, "Solving Elliptic Problems on an Array Processor System," in *Elliptic Problem Solvers II*, ed. A. Schoenstadt, New York, N.Y.: Academic Press, Inc., 1984.

[49] J. T. Schwartz, "Ultracomputers," *ACM TOPLAS*, pp. 484-521, 1980.

[50] H. J. Siegel, *Interconnection Networks for Large-Scale Parallel Processing: Theory and Case Studies*. Lexington, MA: D. C. Heath and Company, 1985.

[51] L. Uhr, "Pyramid Multi-computer Structures and Augumented Pyramids," in *Computing Structures for Image Processing*, ed. M.J.B. Duff, New York, N.Y.: Academic Press, Inc., 1983, pp. 95-112.

[52] C.-L. Wu and T.-Y. Feng, *Tutorial: Interconnection Networks for Parallel and Distributed Processing*. Silver Spring, MD: IEEE Computer Society Press, 1984.

[53] D. M. Young , "Iterative Methods for Solving Partial Differential Equations of Elliptic Type ," Doctoral Thesis , Harvard University , 1950.

PART I : DISCRETIZATION

The first part of this thesis discusses the discretization of boundary value problems with finite-difference methods. A new methodology utilizing the spectral analysis of local differential operators is proposed in Chapter 2 to design and analyze mode-dependent finite-difference schemes for linear homogeneous ordinary and partial differential equations.

We interpret the finite-difference method as a procedure for approximating exactly a local differential operator over a finite-dimensional space of test functions called the coincident space and show that the coincident space is basically determined by the nullspace of the local differential operator. Since local operators are linear and approximately with constant coefficients, we introduce a transform domain approach to perform the spectral analysis. For the case of boundary-value ODEs, a mode-dependent finite-difference scheme can be systematically obtained. For boundary-value PDEs, mode-dependent 5-point, rotated 5-point and 9-point stencil discretizations for the Laplace, Helmholtz and convection-diffusion equations are developed.

The effectiveness of the resulting schemes is shown analytically, as well as by considering several numerical examples.

Chapter 2 : Mode-dependent Finite-difference Method

2.1 Introduction

In order to derive a finite-difference approximation for the derivative of a smooth function, a common procedure is to use a Taylor series to expand the function locally and to select the coefficients such that the order of the discretization error is as high as possible. This procedure is based on the assumption that smooth functions can be well approximated by polynomials locally, and in fact it can be shown that the resulting finite-difference approximation is exact for low order polynomials. However, when the function is exponentially increasing (decreasing) or highly oscillatory, the polynomial representation becomes poor and better finite-difference schemes can be derived if we require that the derivative of exponential or trigonometric functions should be approximated exactly. In our framework, polynomials, exponential and trigonometric functions are all viewed as modes, and finite-difference schemes obtained by an exact approximation of the derivative of a certain number of modes are called *mode-dependent finite-difference* schemes. These modes are the *coincident modes* and the space spanned by them is the *coincident space*.

Historically, the idea of selecting exponential functions as coincident modes was first suggested by Allen and Southwell [1] for discretizing the convection-diffusion equation. An important feature of this problem is that there are large first-order terms in the governing second-order PDE. Due to these large first-order terms, there exists a boundary layer which cannot be well approximated by polynomials. The use of trigonometric functions as coincident modes was first discussed by Gautschi [19] for the numerical integration of ODEs which have periodic or oscillatory solutions whose periods can be estimated in advance. In addition, high order finite-difference schemes for the Laplace equation were derived by choosing some particular polynomials as coincident modes [32].

Although the concept of a mode-dependent finite-difference discretization procedure has been known for years and mentioned repeatedly in literatures (see for examples the references appearing in Section 2.6), few theoretical results about this method have been obtained until now. Important problems, such as whether the mode-dependent finite-difference discretization procedure can always be efficiently applied and how to design such a scheme, remain open. This chapter provides a methodology utilizing the spectral analysis of local differential operators to answer these questions. To avoid unnecessary distractions, we will concentrate on 1D and 2D homogeneous boundary-value problems. However, the general methodology described here also applies to initial value problems as well as nonhomogeneous equations. We will demonstrate this point by referring to some related work.

Since a differential operator is well approximated locally by a linear constant-coefficient operator, the spectral analysis of this local operator becomes relatively easy and a transform domain analysis can be conveniently applied. In the transform domain, the differential and difference operators are algebraic expressions in terms of the complex frequencies s and z. We interpret the mode-dependent finite-difference discretization procedure as a way to specify how these two expressions match each other at a certain number of frequencies in the transform domain. This transform domain viewpoint helps us to gain a better understanding of existing mode-dependent finite-difference schemes and serves as a basis for designing new schemes.

We apply the same methodology to both ODEs and PDEs, and develop several mode-dependent finite-difference schemes. The main results include a $(R+1)$-point mode-dependent central difference scheme for a Rth-order boundary-value ODE, and 5-point, rotated 5-point, 9-point stencil discretizations for the 2D Laplace, Helmholtz and convection-diffusion equations. The mode-dependent finite-difference schemes for the Laplace equation are the same as the conventional ones. However, we present a new derivation. The mode-dependent 5-point and 9-point stencil discretizations of the Helmholtz and convection-diffusion equations are new and have an accuracy proportional to $O(h^2)$ and $O(h^6)$ respectively.

This chapter is organized as follows. In Section 2.2, we describe the mode-dependent finite-difference approximation concept in both the space and transform domains. In Section 2.3, we study the discretization of boundary-value ODEs. The problem of determining the coincident space for homogeneous ODEs is discussed and a mode-dependent finite-difference scheme is presented. This scheme is shown to be exact for constant-coefficient ODEs and has a high degree of accuracy for ODEs with smoothly varying coefficients. The extension to the problem of discretizing nonhomogeneous ODEs is briefly addressed. In Section 2.4, we generalize the methodology from one to two dimensions. In particular, we use the Laplace, Helmholtz and convection-diffusion equations as examples to demonstrate the mode-dependent finite-difference discretization procedure for PDEs. Numerical examples are presented in Section 2.5. Section 2.6 discusses several previous related contributions. The main purpose of this section is to organize the literature concerning the mode-dependent finite-difference approximation so that more examples will be accessible to interested readers. Some generalizations and concluding remarks are given in Section 2.7.

2.2 Mode-dependent Finite-difference Discretization

Consider the class of functions of the form

$$u(x) = \sum_{k=1}^{K} [\, c_{k0} + c_{k1}x + c_{k2}\frac{x^2}{2!} + \cdots + c_{kn_k}\frac{x^{n_k}}{(n_k)!} \,] e^{s_k x} ,$$

where each term $x^p s^{s_k x}$, $0 \leqslant p \leqslant n_k$, is called a mode of order p at the frequency s_k. We are interested in approximating a linear Rth-order constant-coefficient differential operator operating on $u(x)$,

$$L(D) = \sum_{r=0}^{R} a_r D^r ,$$

where $D = \frac{d}{dx}$, by a $(r_2 - r_1 + 1)$-point finite-difference operator

$$L_h(E) = \sum_{r=r_1}^{r_2} b_r E^r ,$$

where E is the shift operator defined on a uniform grid Ω_h with spacing h, i.e. for nh, $(n+r)h \in \Omega_h$, $E^r u(nh) = u((n+r)h)$. L_h corresponds to a forward, backward or central difference operator depending on whether $r_1 = 0$, $r_2 = 0$ or $-r_1 = r_2$, respectively. We use

$$P_n(s) = \{\, u(x): \quad u(x) = e^{sx} \sum_{k=0}^{n} c_k \, x^k \quad \} \tag{1}$$

to represent the space spanned by polynomials of degree at most n multiplied by the factor e^{sx}. A mode-dependent finite-difference discretization scheme is obtained by selecting the coefficients b_r of L_h such that

$$[\, L_h(E) - L(D) \,] u(x) = 0 \qquad \text{for} \quad u(x) \in C \quad \text{and} \quad x \in \Omega_h , \tag{2}$$

where C, called the coincident space of the operator L_h, is the direct sum of subspaces of the form (1), i.e.

$$C = \bigoplus_{k=1}^{K} P_{n_k}(s_k) . \tag{3}$$

A mode in the coincident space C is called a coincident mode, and its frequency is called a coincident frequency.

The mode-dependent finite-difference scheme can be conveniently formulated in the transform domain. We define $L(s)$ by replacing D with s through the use of the Laplace transform in the s-domain,

$$L(s) = \sum_{r=0}^{R} a_r s^r ,$$

while $L_h(z)$ is defined by replacing E with z through the use of the Z-transform in the z-domain,

$$L_h(z) = \sum_{r=r_1}^{r_2} b_r z^r = \sum_{r=r_1}^{r_2} b_r e^{rsh} .$$

where the last equality is due to the fact that since E is related to D via $E = e^{hD}$ [11], we have $z = e^{sh}$. Then, we can define the difference Δ between L and L_h in terms of a single variable s in the transform domain

$$\Delta(s) = L_h(e^{sh}) - L(s) ,$$

and the characterization (2)-(3) of the mode-dependent finite-difference scheme can be equivalently stated in the transform domain as

$$\Delta^{(p)}(s_k) = 0 , \quad 0 \leqslant p \leqslant n_k , \ 1 \leqslant k \leqslant K , \tag{4}$$

where $\Delta^{(n)}(s_k) = \frac{d^n \Delta(s)}{ds^n}\Big|_{s=s_k}$. To check that the characterization (4) is equivalent to (2)-(3), we can proceed as follows. First, using the Taylor series expansion for $\Delta(s)$ around a coincident frequency s_k, we have

$$\Delta(s) = \Delta(s_k) + \Delta^{(1)}(s_k)(s-s_k) + \Delta^{(2)}(s_k)\frac{(s-s_k)^2}{2} + \cdots .$$

In addition, the Laplace transform of $\Delta(D)x^p e^{s_k x}$ is

$$\frac{p!\Delta(s)}{(s-s_k)^{p+1}} = \frac{p!\Delta(s_k)}{(s-s_k)^{p+1}} + \frac{p!\Delta^{(1)}(s_k)}{(s-s_k)^{p}} + \frac{p!\Delta^{(2)}(s_k)}{2(s-s_k)^{p-1}} + \cdots . \tag{5}$$

As a consequence of (4), the leading (n_k+1) terms of (5) vanish, i.e.

$$\frac{p!\Delta(s)}{(s-s_k)^{p+1}} = \frac{\Delta^{(n_k+1)}(s_k)}{(n_k+1)!}(s-s_k)^{n_k-p} + \frac{\Delta^{(n_k+2)}(s_k)}{(n_k+2)!}(s-s_k)^{n_k-p+1} + \cdots .$$

The above function is analytic if and only if $n_k - p \geqslant 0$. So, using the complex inversion formula for the Laplace transform, we find that

$$\Delta(D)x^p e^{s_k x} = 0 \quad \text{for} \quad 0 \leqslant p \leqslant n_k .$$

Since $x^p e^{s_k x}$ with $0 \leqslant p \leqslant n_k$ and $1 \leqslant k \leqslant K$ is a basis of C, the finite-difference approximation is exact for any vector in the space C.

Example 2.1: To illustrate the mode-dependent discretization procedure described above, the coincident modes and coefficients of several 3-point central difference schemes for the first-order derivative D are listed in Table 2.1.

modes	$1, e^{\sigma x}\sin(\omega x), e^{\sigma x}\cos(\omega x)$	$1, e^{\sigma x}, xe^{\sigma x}$ ($\sigma \neq 0$)
b_{-1}	$\dfrac{e^{\sigma h}[\sigma\sin(\omega h)-\omega\cos(\omega h)]+\omega}{2\sin(\omega h)[\cos(\omega h)-\cosh(\sigma h)]}$	$\dfrac{(1-\sigma h)e^{\sigma h}-1}{2h[\cosh(\sigma h)-1]}$
b_0	$\dfrac{\omega\cos(\omega h)\sinh(\sigma h)-\sigma\sin(\omega h)\cosh(\sigma h)}{\sin(\omega h)[\cos(\omega h)-\cosh(\sigma h)]}$	$\dfrac{\sigma h\cosh(\sigma h)-\sinh(\sigma h)}{h[\cosh(\sigma h)-1]}$
b_1	$\dfrac{e^{-\sigma h}[\sigma\sin(\omega h)+\omega\cos(\omega h)]-\omega}{2\sin(\omega h)[\cos(\omega h)-\cosh(\sigma h)]}$	$\dfrac{-(1+\sigma h)e^{-\sigma h}+1}{2h[\cosh(\sigma h)-1]}$

modes	$1, e^{\sigma_1 x}, e^{\sigma_2 x}$ ($\sigma_1 \neq \sigma_2$)	$1, x, x^2$
b_{-1}	$\dfrac{1}{2}\dfrac{\sigma_2(1-e^{\sigma_1 h})-\sigma_1(1-e^{\sigma_2 h})}{\sinh[(\sigma_2-\sigma_1)h]+\sinh(\sigma_1 h)-\sinh(\sigma_2 h)}$	$\dfrac{-1}{2h}$
b_0	$\dfrac{\sigma_2\sinh(\sigma_1 h)-\sigma_1\sinh(\sigma_2 h)}{\sinh[(\sigma_2-\sigma_1)h]+\sinh(\sigma_1 h)-\sinh(\sigma_2 h)}$	0
b_1	$\dfrac{1}{2}\dfrac{\sigma_1(1-e^{-\sigma_2 h})-\sigma_2(1-e^{-\sigma_1 h})}{\sinh[(\sigma_2-\sigma_1)h]+\sinh(\sigma_1 h)-\sinh(\sigma_2 h)}$	$\dfrac{1}{2h}$

Table 2.1: The approximation of the first-order derivative D ($L(s) = s$) by a central mode-dependent finite-difference scheme.

2.3 Discretization of Boundary-value ODEs

2.3.1 Homogeneous ODEs

Consider an R th-order homogeneous two-point boundary value problem on [0,1] with Dirichlet boundary conditions. For convenience, we consider the case $R=2m$. The case $R=2m+1$ gives rise to a similar analysis. The equation is written as

$$L\ u = 0\,, \qquad \text{where } L = \sum_{r=0}^{2m} a_r(x)D^r \ \text{ and } \ a_{2m}(x) = 1\,, \tag{1}$$

with given $u(0)$ and $u(1)$. We discretize (1) on a uniform grid with spacing h by a $(2m+1)$-point central difference scheme,

$$L_h\ u_h = 0\,, \qquad \text{where} \quad L_h = \sum_{r=-m}^{m} b_r(nh)E^r\,, \tag{2}$$

and u_h is the estimate of u on grid points. Suppose that ϕ is an arbitrary function in the nullspace N_L of operator L, and that N_L is contained in the coincident space C of L_h. Then, since

$$L\ \phi = 0 \quad \text{and} \quad [\,L_h - L\,]\,\phi = 0\,,$$

we obtain

$$L_h\ \phi = 0\,. \tag{3}$$

Since the discretization for an arbitrary function in the nullspace N_L is exact, we conclude that equation (1) is exactly discretized by (2).

The nullspace N_L is easy to find if the coefficients $a_r(x)$ of L are constant. Even if these coefficients are not constant but smoothly-varying, L still can be well approximated by a constant-coefficient operator in a local region. This simplification is always assumed for finite-difference schemes since the finite-difference method is a local approximation method. For convenience, we drop the spatial dependency of coefficients $a_r(x)$ and $b_r(x)$, and use the notation $a_r \approx a_r(x_0)$ and $b_r \approx b_r(x_0)$ inside operators L and L_h for the rest of this chapter. If $a_r(x)$ and $b_r(x)$ are spatially varying, the discussion is understood to be a *local* analysis in the neighborhood of x_0.

The spectral analysis of a linear constant-coefficient operator $L = \sum_{r=0}^{2m} a_r D^r$ can be easily performed in the transform domain. We choose the coincident frequencies corresponding to modes in the nullspace of L to be roots of the characteristic equation,

$$L(s) = s^{2m} + a_{2m-1} s^{2m-1} + \cdots + a_1 s + a_0 = 0 .$$

In general, $L(s)$ can be factored as

$$L(s) = \prod_{k=1}^{K} (s - s_k)^{n_k} , \qquad \text{where} \quad \sum_{k=1}^{K} n_k = 2m ,$$

and s_k is known as a *natural frequency* of L of order n_k. As a consequence, the operator L has the $2m$-dimensional nullspace

$$N_L = \bigoplus_{k=1}^{K} P_{n_k-1}(s_k) .$$

A $(2m+1)$-point finite difference scheme can be uniquely determined by a $(2m+1)$-dimensional coincident space C. However, since a homogeneous finite-difference equation such as (2) can be scaled arbitrarily, a $2m$-dimensional coincident space C is sufficient to specify L_h in (2). So, letting

$$C = N_L ,$$

we have an exact discretization scheme for (1). For this choice, L_h can be determined easily as

$$L_h(z) = A\, z^{-m} \prod_{k=1}^{K} (z - z_k)^{n_k} , \qquad \text{where} \quad z_k = e^{s_k h} , \tag{4}$$

where A is a scaling factor and the multiplication factor z^{-m} is due to the fact that we want $L_h(z)$ to be a central difference scheme. This can be verified by substituting $L(s)$ and $L_h(e^{sh})$ back into (2.2.4).

Hence, after inverse transformation, we obtain the following mode-dependent finite-difference scheme for (1) in the space domain

$$L_h u_h = 0\,, \quad \text{where} \quad L_h(E) = A\ E^{-m} \prod_{k=1}^{K} (E - e^{s_k h})^{n_k}\,, \tag{5}$$

and s_k is a natural frequency of L of order n_k. The scaling factor A does not affect the solution of the system of equations (5). However, in order to analyze the discretization error $\Delta(s)$ appropriately, it is important to choose A such that $L_h(e^{sh})$ and $L(s)$ are consistent over fine grids. This consideration requires that the scaling factor A of (5) should be proportional to $\frac{1}{h^{2m}}$, as h goes to zero.

Example 2.2: (1D Laplace equation) For $L(D) = D^2$, we know that $N_L = \{1, x\}$. The coincident modes have the same frequency $s_k = 0$. According to (5), we have

$$L_h(E) = A\ E^{-1}(E-1)^2 = A\ (E - 2 + E^{-1})\,. \tag{6}$$

If we choose $C = N_L + \{x^2\}$, the constant A can be uniquely determined. Solving $Lx^2 = L_h x^2$, where $x \in \Omega_h$, we find that $A = \frac{1}{h^2}$. Then, (6) reduces to the standard 3-point central difference scheme.

Example 2.3: (1D convection-diffusion equation) The differential operator is $L(D) = D^2 - a_1 D$, where $a_1 \neq 0$. In this case, $N_L = \{1, e^{a_1 x}\}$ and $s_k = 0, a_1$. Therefore, by (5), we have

$$L_h(E) = A\ E^{-1}(E-1)(E - e^{a_1 h}) = A\ [E - (1+e^{a_1 h}) + e^{a_1 h} E^{-1}]\,. \tag{7}$$

In particular, if $C = N_L + \{x\}$, we find that $A = \frac{a_1}{h(e^{a_1 h} - 1)}$. Then, (7) is identical to the scheme considered by Allen and Southwell [1].

For comparison, consider the conventional finite-difference scheme for (1),

$$L_{h,c}u_h = 0\,, \quad \text{where} \quad L_{h,c}(E) = \sum_{r=0}^{2m} a_r h^{-r} D_{h,2m+1}^r(E)\,, \tag{8}$$

where $h^{-r} D_{h,2m+1}^r(E)$ denotes the $(2m+1)$-point central difference operator for the rth-order derivative D^r which is obtained by selecting $C = P_{2m}(0)$ as coincident space, and by requiring consistency over fine grids. Then, by comparing (5) and (8), we see that the mode-dependent scheme (5) is obtained by discretizing term by term the product form of the differential operator $L(D)$, whereas the conventional scheme (8) is found by discretizing term by term the summation form of $L(D)$.

According to the above discussion, the approximation of the differential operator $L(D)$ in (1) by $L_h(E)$ given by (5) does not give rise to any discretization error when the coefficients a_r are constant. This fact is also supported by numerical results.

Of course, the mode-dependent scheme (5) gives rise to a discretization error when the coefficients a_r are spatially varying. This discretization error depends on the smoothness of the ODE coefficients and the grid size h. However, the exact form of this dependency is still unknown, and we have yet to develop a general procedure for estimating the size of the error in this case. In Section 2.5, we use a 1D convection-diffusion equation as a test problem and find that the error of the mode-dependent scheme is proportional to $O(\epsilon h^2)$, where ϵ is the first order derivative of the coefficient function, while that of the conventional scheme is proportional to $O(h^2)$ [11]. The mode-dependent scheme is always better than the conventional central-difference scheme in this test problem and the improvement in accuracy offorded by the mode-dependent scheme becomes larger as the coefficient of the convection-diffusion equation becomes smoother.

2.3.2 Extensions to Nonhomogeneous ODEs

Suppose that (1) includes a driving function $f(x)$, so that

$$L u - f = 0 . \tag{9}$$

By performing a Taylor series expansion of $f(x)$ in the vicinity of a discretization point x_0, we can assume that f is approximated locally by a polynomial of low degree, i.e.

$$f(x) \approx c_0 + c_1 x + c_2 x^2 + \cdots .$$

A general discretization scheme for (9), which has been proposed in the context of the OCI and HODIE methods [4][7][31], is

$$L_h u_h - I_h f = L_h u_h - I_h Lu = 0 , \tag{10}$$

where I_h is an averaging operator.

The set of functions whose images through L are polynomials of degree less or equal to l defines the space

$$P_{L,l} = \{ u(x) : Lu = \sum_{r=0}^{l} p_r x^r \} .$$

Note that since the coefficients p_r above can all be selected equal to zero, N_L is also included in $P_{L,l}$. The space $P_{L,l}$ will be used here to approximate the solution space of equation (9). Suppose that ψ is an arbitrary function of the space $P_{L,l}$. Ideally, we want

$$L_h \psi - I_h L \psi = 0 , \tag{11}$$

in order to guarantee that the discretization (10) of the nonhomogeneous equation (9) is exact in the approximated solution space $P_{L,l}$.

In particular, if I_h is chosen to be the identity operator I, (11) becomes

$$(L_h - L)\psi = 0 . \tag{12}$$

Therefore, the coincident space C of the finite-difference operator L_h for the nonhomogeneous equation (9) has to be

$$C = P_{L,l} .$$

The major disadvantage of this choice is that the dimension of C is larger than that of N_L. Hence, a finite-difference method with more than $(2m+1)$-points will be necessary and more computations will be required.

The purpose of introducing I_h is to reduce the dimension of the coincident space. For a $(2m+1)$-point finite-difference scheme, we can decompose the discretization scheme (10) into two steps. First, by choosing $C = P_{L,0}$, we can uniquely determine L_h. Then, by using an arbitrary function ψ of the space $P_{L,l} \ominus P_{L,0}$ as a test function for (11), we can solve for the coefficients of I_h. This procedure is illustrated by the following example.

Example 2.4: (1D Poisson equation) In this case, we have $L(D) = D^2$, $N_L = \{ 1, x \}$, and $P_{L,l} = \{ 1, x, x^2, x^3, \cdots, x^{l+2} \}$. By choosing $C = P_{L,0}$, we know from example 2 that

$$L_h(E) = \frac{1}{h^2}(E - 2 + E^{-1}) .$$

Assuming that $I_h(E) = d_{-1}E^{-1} + d_0 + d_1 E$ with respect to the same grid Ω_h and solving (11) with L_h given above and $\psi = x^3, x^4$, we obtain

$$I_h(E) = \frac{1}{12}E^{-1} + \frac{5}{6} + \frac{1}{12}E .$$

Then, the discretization (11) is exact for any function in the space $P_{L,2}$. More generally, we call $P_{L,l}$ the *generalized coincident space* $C_g(L_h, I_h)$ for the approximation (10) of (9). Note that the dimension of C_g for the above example is 5, and that there are 5 independent parameters in (L_h, I_h) since (11) can be scaled by an arbitrary constant.

The above approach is different from the HODIE method. For an Rth-order nonhomogeneous ODE, the HODIE method uses polynomials of degree less or equal to n, i.e. $P_n(0)$ with $n > R$, as the generalized coincident space C_g for equation (10). It does not exploit any special structure of the differential operator L. In contrast, our mode-dependent method uses the approximated solution space $P_{L,n-R}$ as the generalized coincident space C_g. Hence, a spectral analysis of the operator L is necessary. In particular, when $L = D^R$, $P_{L,n-R}$ is the same as $P_n(0)$. Then, there is no difference between the HODIE and mode-dependent methods.

The determination of the averaging operator I_h for the HODIE method has been discussed in detail [7][31]. For example, the operator $I_{h'}$ may be defined on an *auxiliary* grid $\Omega_{h'}$ different from the discretization grid Ω_h. A similar approach can also be used to design I_h for the mode-dependent method. Note that the selection of the averaging operator I_h has no effect on functions in the nullspace N_L. Therefore, the coincident space C of L_h has to contain N_L so that the discretization error for functions in N_L can be eliminated by choosing an appropriate L_h.

In this chapter, we focus primarily on the determination of the coincident space C and of the finite-difference operator L_h. In the next section, we will therefore restrict our attention to *homogeneous* boundary-value PDEs and we will attempt to extend the methodology developed in this section to the discretization of this specific class of PDE problems.

2.4 Discretization of Boundary-value PDEs

Consider a general two-dimensional boundary-value PDE on the square $[0,1]^2$

$$L(D_x, D_y)\, u = 0\,, \quad \text{where } L(D_x, D_y) = \sum_{r,s} a_{r,s} D_x^r D_y^s\,, \tag{1}$$

$D_x^r = \dfrac{\partial^r}{\partial x^r}$ and $D_y^s = \dfrac{\partial^s}{\partial y^s}$, with Dirichlet boundary conditions. We discretize (1) with the finite-difference scheme

$$L_{h_x,h_y}(E_x, E_y)\, u_{h_x,h_y} = 0\,, \qquad \text{where} \qquad L(E_x, E_y) = \sum_{r,s} b_{r,s} E_x^r E_y^s\,, \tag{2}$$

and where E_x and E_y are respectively the shift operators in the x- and y-directions on the grid Ω_{h_x,h_y}. Relying on a natural generalization of the 1D case, we have the following associations between the 2D space domain operators and transform domain variables

$$D_x \longleftrightarrow s_x \quad , \quad D_y \longleftrightarrow s_y \quad , \quad E_x \longleftrightarrow z_x \quad , \quad E_y \longleftrightarrow z_y \quad .$$

where $s_x = \sigma_x + i\,\omega_x$ and $s_y = \sigma_y + i\,\omega_y$. They are related via $E_x = e^{h_x D_x}$, $E_y = e^{h_y D_y}$, $z_x = e^{h_x s_x}$ and $z_y = e^{h_y s_y}$. To simplify the following discussion, we will only consider the case $h_x = h_y = h$.

Substituting $e^{s_x x + s_y y}$ inside (1), we obtain the characteristic equation

$$\sum_{r,s} a_{r,s}\, s_x^r s_y^s = 0\,. \tag{3}$$

There are two complex variables in (3), but since we have only one (complex) equation, there are uncountably infinitely many solutions to this equation and therefore infinitely many potential modes in N_L. It is not possible to approximate all modes in N_L exactly. Thus, we have to select a finite-dimensional subspace $D_L \subset N_L$, called the dominant-mode space, as the coincident space C for L_h. The determination of D_L

depends on a rough estimation of the local behavior of the solution. This information is usually provided by the structure of the PDE operator that we consider and the corresponding boundary conditions. In this section, we restrict our attention to the case where the dominant modes are either oscillating or exponentially growing (decaying). In other words, coincident frequencies are selected among the sets

$$\{ (s_x, s_y) : (s_x, s_y) = (\sigma_x, \sigma_y) \} \quad or \quad \{ (s_x, s_y) : (s_x, s_y) = (i\,\omega_x, i\,\omega_y) \} . \qquad (4)$$

We do not consider complex coincident frequencies, since they generally lead to discretization schemes with complex coefficients which complicate the solution procedure. However, even under (4), the mode-dependent concept still does not lead to a unique discretization scheme. By taking into account the symmetrical property of the spectra of the differential and difference operators and the solubility of the resulting finite-difference schemes, we can further constrain ourselves within a much smaller design space. In the following, the 5-point, rotated 5-point and 9-point stencil discretizations for the Laplace, Helmholtz and convection-diffusion equations will be used as examples to demonstrate the mode-dependent discretization concept.

2.4.1 Laplace Equation

For the Laplace equation, we have $L(D_x, D_y) = D_x^2 + D_y^2$. Since only one frequency $(s_x, s_y) = (0,0)$ satisfies the characteristic equation and belongs to the sets (4) of interest, (0,0) is selected as the unique coincident frequency. In this case, the mode-dependent scheme is the same as the conventional scheme.

The following 5-point, rotated 5-point and 9-point stencil discretization schemes have been derived by several approaches [11][25][32],

$$L_{h,+}(E_x,E_y) = \frac{1}{h^2}\left(E_x^{-1} + E_x + E_y^{-1} + E_y - 4 \right), \tag{5}$$

$$L_{h,\times}(E_x,E_y) = \frac{1}{2h^2}\left(E_x^{-1}E_y^{-1} + E_x^{-1}E_y + E_xE_y^{-1} + E_xE_y - 4 \right), \tag{6}$$

$$L_{h,9} = \frac{1}{6h^2}\left[4(E_x + E_x^{-1} + E_y + E_y^{-1}) + (E_xE_y + E_x^{-1}E_y + E_xE_y^{-1} + E_x^{-1}E_y^{-1}) - 20 \right]. \tag{7}$$

It is well known that they have respectively an accuracy of $O(h^2)$, $O(h^2)$ and $O(h^6)$ when used to discretize the Laplace equation [25].

Here, we present another derivation of these schemes by matching $L(s_x,s_y)$ and $L_h(z_x,z_y)$ at the coincident frequency (0,0) in the transform domain. As before, we consider the expansion of $\Delta = L_h - L$ around (0,0),

$$\Delta(s_x,s_y) = \Delta^{(0,0)}(0,0) + \Delta^{(1,0)}(0,0)s_x + \Delta^{(0,1)}(0,0)s_y + \Delta^{(2,0)}(0,0)s_x^2$$
$$+ \Delta^{(1,1)}(0,0)2s_xs_y + \Delta^{(0,2)}(0,0)s_y^2 + \sum_{\substack{p+q\geqslant 3\\ p,q\geqslant 0}} \Delta^{(p,q)}(0,0)\frac{(p+q)!}{p!q!}s_x^ps_y^q, \tag{8}$$

where

$$\Delta^{(p,q)}(0,0) = \frac{\partial^{p+q}\Delta(s_x,s_y)}{\partial^p s_x\,\partial^q s_y}\Big|_{(s_x,s_y)=(0,0)},$$

which is a function of the grid size h. Hence, (8) is in fact a power series expansion in h. Our derivation attempts to make the order of the residual terms in (8) as high as possible.

The discretization schemes (5) and (6) can be derived by imposing the requirement that L and L_h should be consistent over fine grids, and by requiring respectively that

$$\Delta^{(0,0)}(0,0) = 0 \ , \ \Delta^{(1,0)}(0,0) = 0 \ , \ \Delta^{(0,1)}(0,0) = 0 \ , \ \Delta^{(2,0)}(0,0) = \Delta^{(0,2)}(0,0) \ ,$$

and

$$\Delta^{(0,0)}(0,0) = 0 \ , \ \Delta^{(1,0)}(0,0) = 0 \ , \ \Delta^{(0,1)}(0,0) = 0 \ , \ \Delta^{(1,1)}(0,0) = 0 \ .$$

Note that if $\Delta^{(2,0)}(0,0) = \Delta^{(0,2)}(0,0)$, the fourth and sixth terms of the right-hand side of (7) can be combined as $\Delta^{(2,0)}(0,0)(s_x^2+s_y^2)$. This term becomes equal to zero if we restrict our attention in (7) to the frequencies (s_x, s_y) which satisfy the characteristic equation $s_x^2+s_y^2=0$ of the Laplacian operator. Similarly, the 9-point stencil formula (7) can be derived by requiring

$$\Delta^{(0,0)}(0,0) = 0 \ , \ \Delta^{(1,0)}(0,0) = 0 \ , \ \Delta^{(0,1)}(0,0) = 0 \ , \ \Delta^{(2,0)}(0,0) = \Delta^{(0,2)}(0,0) \ ,$$
$$\Delta^{(1,1)}(0,0) = 0 \ , \ \Delta^{(1,2)}(0,0) = 0 \ , \ \Delta^{(2,1)}(0,0) = 0 \ , \ \Delta^{(4,0)}(0,0) = 3\,\Delta^{(2,2)}(0,0) \, .$$

2.4.2 Helmholtz Equation

For the Helmholtz equation, we have

$$L(D_x, D_y) = D_x^2 + D_y^2 + \lambda^2 \, .$$

If s_x and s_y are purely imaginary, the characteristic equation becomes

$$\omega_x^2 + \omega_y^2 = \lambda^2 \, , \tag{9}$$

which is a circle in the ω_x-ω_y plane, centered at the origin and with radius $|\lambda|$. There are infinitely many natural frequencies and, hence, there are many different ways to select coincident frequencies. In this section, we design mode-dependent 5-point, rotated 5-point and 9-point stencil discretization schemes based on the following two considerations. First, if there is no further information about the dominant modes, a reasonable strategy is to distribute coincident frequencies uniformly along the contour (9). Second, we want to preserve the symmetry properties of L so that the resulting discretization scheme is in a simple form and can easily be implemented.

Let us select

$$(|\lambda| \cos(\tfrac{n}{2}\pi+\tfrac{1}{4}\pi)i \ , \ |\lambda| \sin(\tfrac{n}{2}\pi+\tfrac{1}{4}\pi)i \ ), \qquad 0 \leqslant n \leqslant 3 ,$$

as coincident frequencies as shown in Figure 2.1(a). With this choice, the discretization along the x - and y -directions can be treated independently. The resulting scheme is

$$L_h(E_x, E_y) = A \ [E_x^{-1} - 2\cos(\frac{|\lambda|}{\sqrt{2}}h) + E_x + \kappa(E_y^{-1} - 2\cos(\frac{|\lambda|}{\sqrt{2}}h) + E_y)] .$$

Two parameters A and κ remain undetermined in the above expression. The parameter κ is selected such that the discretization error $\Delta(s_x, s_y)$ at natural frequencies is proportional to $O(h^2)$, and the parameter A is used to normalize the above scheme so that L_h is consistent with L. A simple choice of κ and A for the Helmholtz equation is $\kappa = 1$ and $A = \frac{1}{h^2}$. Hence, this gives a symmetrical 5-point stencil discretization operator

$$L_{h,+}(E_x, E_y) = \frac{1}{h^2} [E_x^{-1} + E_x + E_y^{-1} + E_y - 4\cos(\frac{|\lambda|}{\sqrt{2}}h)] . \tag{10}$$

Rotating the above four coincident frequencies in the transform domain and the above 5-point stencil in the space domain by an angle $\frac{1}{4}\pi$, we obtain another mode-dependent 5-point stencil discretization. In this scheme, the coincident frequencies become

$$(|\lambda| \cos(\tfrac{n}{2}\pi)i \ , \ |\lambda| \sin(\tfrac{n}{2}\pi)i \ ), \qquad 0 \leqslant n \leqslant 3 .$$

as shown in Figure 2.1(b), and the resulting 5-point stencil operator is

$$L_{h,\times}(E_x, E_y) = \frac{1}{2h^2} [E_x^{-1}E_y^{-1} + E_x^{-1}E_y + E_x E_y^{-1} + E_x E_y - 4\cos(|\lambda| h)] . \tag{11}$$

Notice that this rotated 5-point stencil can be viewed as corresponding to a discretization scheme on a grid with spacing $\sqrt{2}h$. By appropriately combining (10), (11) and adding a constant term, we obtain the 9-point stencil discretization operator,

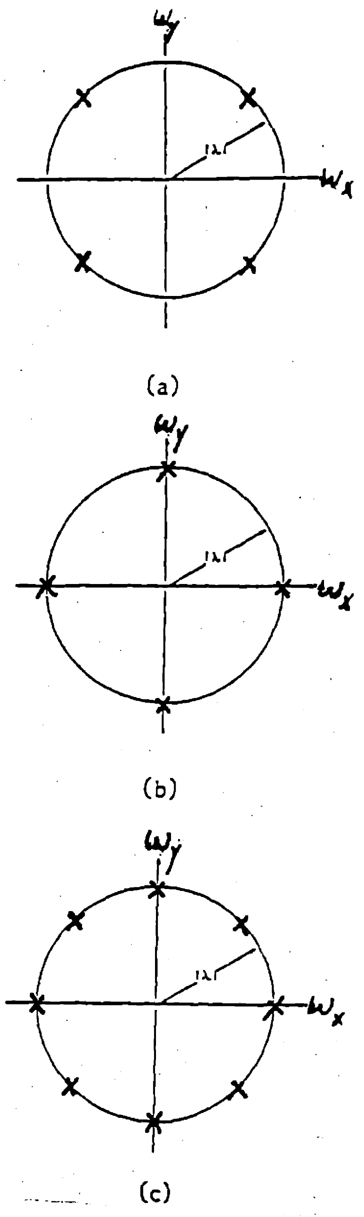


Figure 2.1: Coincident frequencies of the mode-dependent (a) 5-point (b) rotated 5-point and (c) 9-point stencils discretizations of the Helmholtz equation.

$$L_{h,9}(E_x, E_y) = \frac{\gamma_\times}{\gamma_\times+\gamma_+} L_{h,+}(E_x, E_y) + \frac{\gamma_+}{\gamma_\times+\gamma_+} L_{h,\times}(E_x, E_y) - \frac{\gamma_\times \gamma_+}{\gamma_\times+\gamma_+} . \qquad (12)$$

Then, if

$$\gamma_\times = L_{h,\times}(e^{i\frac{|\lambda|}{\sqrt{2}}h}, e^{i\frac{|\lambda|}{\sqrt{2}}h}) = \frac{1}{h^2}[\cos(\sqrt{2}|\lambda|h) + 1 - 2\cos(|\lambda|h)],$$

$$\gamma_+ = L_{h,+}(e^{i|\lambda|h}, 1) = \frac{1}{h^2}[2\cos(|\lambda|h) + 2 - 4\cos(\frac{|\lambda|}{\sqrt{2}}h)],$$

we are able to match $L_h(z_x, z_y)$ and $L(s_x, s_y)$ at 8 frequencies

$$(|\lambda|\cos(\frac{n}{4}\pi)i \ , \ |\lambda|\sin(\frac{n}{4}\pi)i \ ), \qquad 0 \leqslant n \leqslant 7 .$$

as shown in Figure 2.1(c). Thus, (12) is the desired mode-dependent 9-point stencil discretization operator.

When $|\lambda|$ goes to zero, the Helmholtz equation reduces to the Laplace equation and schemes (10)-(12) converge to (5)-(7). So, schemes (10)-(12) can be viewed as a natural generalization of (5)-(7) and apply to both $\lambda=0$ and $\lambda\neq0$. The error estimate of the above schemes for the test functions $e^{s_x x + s_y y}$, where s_x and s_y satisfy the characteristic equation $s_x^2 + s_y^2 + \lambda^2 = 0$, can obtained in a straightforward way. Since

$$\Delta(D_x, D_y)e^{s_x x + s_y y} = L_h(e^{D_x h}, e^{D_y h})e^{s_x x + s_y y} = L_h(e^{s_x h}, e^{s_y h})e^{s_x x + s_y y} ,$$

we only have to replace E_x and E_y with $e^{s_x h}$ and $e^{s_y h}$ inside $L_h(E_x, E_y)$ and use a Taylor series expansion to simplify the resulting algebraic equation, which is a power series of h and the leading term is defined as the accuracy of the corresponding discretization scheme. By using this approach, we find that the two 5-point stencil discretization schemes (10), (11) and the 9-point stencil scheme (12) have an accuracy of $O(h^2)$, $O(h^2)$ and $O(h^6)$ respectively (see the appendix on page 70).

Unlike in the ODE case, the mode-dependent schemes for PDEs cannot catch all modes in the null space of L, so that there exist discretization errors even for constant-coefficient PDEs. Rigorously speaking, the above error estimate applies only to constant-coefficient PDEs. If the coefficients of the PDE of interest are spatially varying, the error associated to mode-dependent schemes is still unknown. But, we suspect that when the coefficients are smoothly varying, the error is approximately the same as for constant coefficients.

Conventional finite-difference schemes for the Helmholtz equation are derived by discretizing the Laplacian with operators (5)-(7) and then combining them with the remaining term $\lambda^2 u$. The resulting schemes have all an accuracy of $O(h^2)$. Therefore, the conventional 9-point discretization scheme is much worse than the mode-dependent 9-point scheme. Although the conventional and mode-dependent 5-point schemes have the same order of accuracy, the mode-dependent schemes (10) and (11) are more accurate than conventional schemes along the contour (9). To show this, the discretization errors for mode-dependent and conventional 5-point discretization schemes are plotted in Figure 2.2 for the case where $\lambda = 10$ and $h = 0.1$.

On the other hand, we can also consider the discretization of the operator

$$\tilde{L}(D_x, D_y) = D_x^2 + D_y^2 - \lambda^2 .$$

Considering only the real frequencies $(s_x, s_y) = (\sigma_x, \sigma_y)$, we have the characteristic equation,

$$\sigma_x^2 + \sigma_y^2 = \lambda^2 .$$

Thus, for the present case, we examine the $\sigma_x - \sigma_y$ plane, instead of the $\omega_x - \omega_y$ plane.

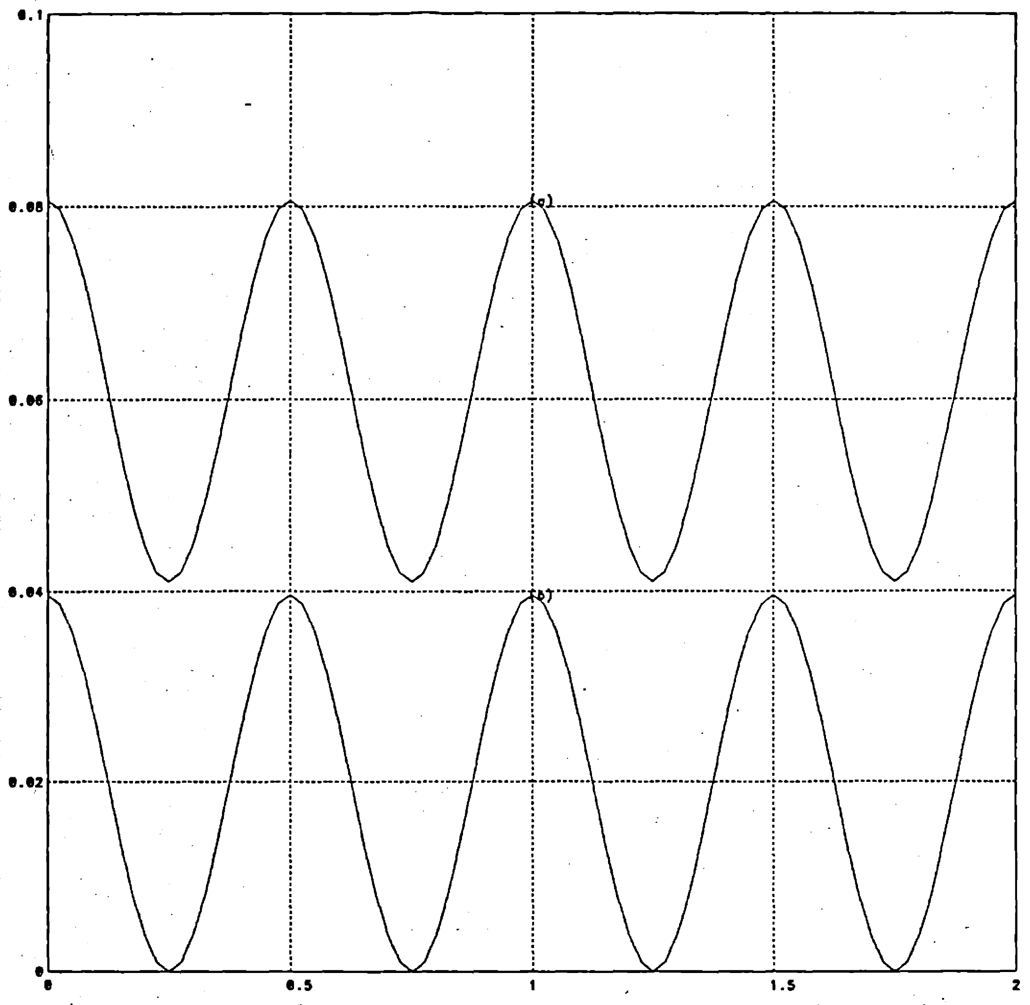


Figure 2.2: Plot of $|\Delta(\omega_x, \omega_y)|$ as a function of c along the contour $(\omega_x, \omega_y) = (|\lambda|\cos(c\pi), |\lambda|\sin(c\pi))$, for (a) conventional and (b) mode-dependent 5-point stencil discretizations of the Helmholtz equation.

By using an approach similar to the one described above, we get the following 5-point and 9-point stencil discretization schemes

$$\tilde{L}_{h,+}(E_x, E_y) = \frac{1}{h^2}\,[\, E_x^{-1} + E_x + E_y^{-1} + E_y - 4\cosh(\frac{|\lambda|}{\sqrt{2}}h)\,]\,,$$

$$\tilde{L}_{h,\times}(E_x, E_y) = \frac{1}{2h^2}\,[\, E_x^{-1}E_y^{-1} + E_x^{-1}E_y + E_x E_y^{-1} + E_x E_y - 4\cosh(|\lambda| h)\,]\,,$$

$$\tilde{L}_{d,9}(E_x, E_y) = \frac{\gamma_\times}{\gamma_\times+\gamma_+}\tilde{L}_{h,+}(E_x, E_y) + \frac{\gamma_+}{\gamma_\times+\gamma_+}\tilde{L}_{h,\times}(E_x, E_y) - \frac{\gamma_\times\gamma_+}{\gamma_\times+\gamma_+}\,,$$

where

$$\gamma_\times = \frac{1}{h^2}\,[\,\cosh(\sqrt{2}|\lambda| h) + 1 - 2\cosh(|\lambda| h)\,],\quad \gamma_+ = \frac{1}{h^2}\,[\,2\cosh(|\lambda| h) + 2 - 4\cosh(\frac{|\lambda|}{\sqrt{2}}h)\,]\,.$$

These schemes have an accuracy of $O(h^2)$, $O(h^2)$ and $O(h^6)$ respectively.

2.4.3 Convection-diffusion Equation

For the convection-diffusion equation, the differential operator takes the form

$$L(D_x, D_y) = D_x^2 + D_y^2 - 2\alpha D_x - 2\beta D_y\ .$$

In particular, if we consider only real frequencies $(s_x, s_y) = (\sigma_x, \sigma_y)$, the corresponding characteristic equation is

$$\sigma_x^2 + \sigma_y^2 - 2\alpha\sigma_x - 2\beta\sigma_y = 0\,, \tag{13}$$

which is a circle in the σ_x-σ_y plane centered at $(\alpha\,,\beta)$ with radius $d = \sqrt{\alpha^2+\beta^2}$.

The conventional approach for discretizing the above equation relies on a central difference scheme to approximate the first and second order derivatives separately. This gives

$$L_{h,c}(E_x, E_y) = \frac{1}{h^2}\,[\,(1+\alpha h)E_x^{-1} + (1-\alpha h)E_x - 4 + (1+\beta h)E_y^{-1} + (1-\beta h)E_y\,]\,, \tag{14}$$

which corresponds to selecting a single coincident frequency at the origin. Allen and Southwell combined two 1D mode-dependent schemes along the x- and y-directions

[1] (also see example 3 in Section 3). This leads to

$$L_{h,AS}(E_x, E_y) = \frac{1}{h}\left[\frac{2\alpha}{e^{2\alpha h}-1}(e^{2\alpha h}E_x^{-1} - 1 - e^{2\alpha h} + E_x) + \frac{2\beta}{e^{2\beta h}-1}(e^{2\beta h}E_y^{-1} - 1 - e^{2\beta h} + E_y)\right]. \qquad (15)$$

which corresponds to selecting (0,0), $(2\alpha,0)$, $(0,2\beta)$, $(2\alpha,2\beta)$ as coincident frequencies. Motivated by the discussion in the previous section, we select the coincident frequencies

$$(\ \alpha + d\cos(\frac{n}{2}\pi + \frac{1}{4}\pi)\ ,\ \beta + d\sin(\frac{n}{2}\pi + \frac{1}{4}\pi)\ )\ , \qquad 0 \leqslant n \leqslant 3\ ,$$

uniformly along the contour (13) and obtain the following stencil

$$L_{h,+}(E_x, E_y) = \frac{1}{h^2}\,[\,e^{\alpha h}E_x^{-1} + e^{-\alpha h}E_x + e^{\beta h}E_y^{-1} + e^{-\beta h}E_y - 4\cosh(\frac{d}{\sqrt{2}}h\ )\,]\,. \quad (16)$$

The multiplication of E_x by the factor $e^{-\alpha h}$ in the x-direction of the space domain corresponds to a shift in the s_x-coordinate in the transform domain, where s_x becomes $s_x - \alpha$, and a similar argument applies also to the y-direction. Therefore, the above scheme in fact shifts the center (α,β) of the circle (13) back to the origin and treats it as the Helmholtz equation with radius d. The coincident frequencies of these three schemes are plotted in Figure 2.3. Although all schemes have an accuracy of $O(h^2)$, schemes (15) and (16) are always diagonally dominant while the conventional scheme (14) loses this property for large cell Reynolds numbers αh and βh. This is one major disadvantage associated with the conventional central difference scheme.

Following a similar procedure, we can also design mode-dependent rotated 5-point and 9-point stencil discretization schemes for the convection-diffusion equation. This gives

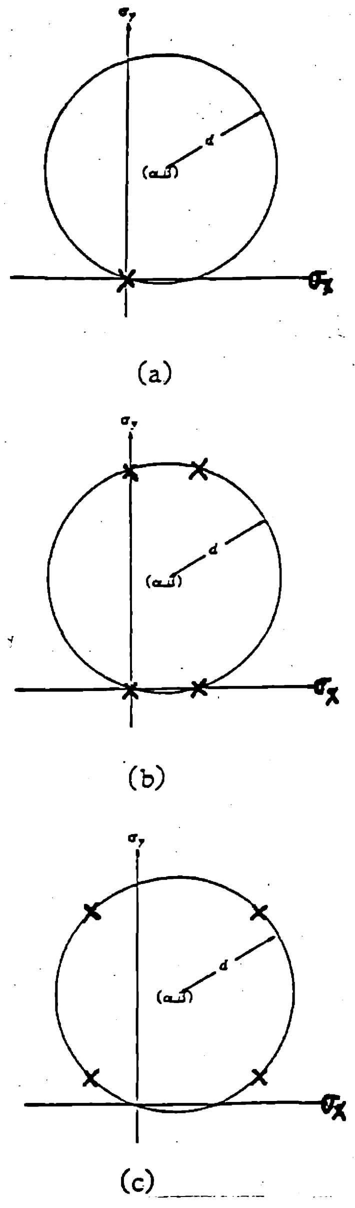


Figure 2.3: Coincident frequencies of the (a) central difference, (b) Allen-Southwell, and (c) uniformly-distributed mode-dependent 5-point stencil discretizations of the convection-diffusion equation.

$$L_{h,\times}(E_x, E_y) = \frac{1}{2h^2}[\, e^{(\alpha+\beta)h} E_x^{-1} E_y^{-1} + e^{(\alpha-\beta)h} E_x^{-1} E_y$$
$$+ e^{(-\alpha+\beta)h} E_x E_y^{-1} + e^{-(\alpha+\beta)h} E_x E_y - 4\cosh(dh)\,]\,, \qquad (17)$$

$$L_{h,9}(E_x, E_y) = \frac{\gamma_\times}{\gamma_\times+\gamma_+} L_{h,+}(E_x, E_y) + \frac{\gamma_+}{\gamma_\times+\gamma_+} L_{h,\times}(E_x, E_y) - \frac{\gamma_\times \gamma_+}{\gamma_\times+\gamma_+}\,, \qquad (18)$$

with

$$\gamma_\times = \frac{1}{h^2}[\, \cosh(\sqrt{2}dh) + 1 - 2\cosh(dh)\,],\ \ \gamma_+ = \frac{1}{h^2}[\, 2\cosh(dh) + 2 - 4\cosh(\frac{d}{\sqrt{2}}h)\,]\,.$$

These schemes have an accuracy of $O(h^2)$ and $O(h^6)$ respectively.

Appendix to Section 2.4: Derivation of the accuracy of the 5-point, rotated 5-point and 9-point stencil discretizations.

In this appendix, we show analytically that $L_{h,+}$, $L_{h,\times}$ and $L_{h,9}$ in (10), (11) and (12) have an accuracy of $O(h^2)$, $O(h^2)$ and $O(h^6)$ for the test functions $e^{s_x x + s_y y}$ where s_x and s_y satisfy

$$s_x^2 + s_y^2 = -\lambda^2 . \tag{A1}$$

The following equalities can be derived directly from (A1),

$$s_x^4 + s_y^4 = \lambda^4 - 2s_x^2 s_y^2 , \qquad (s_x + s_y)^4 + (s_x - s_y)^4 = 2\lambda^4 + 8s_x^2 s_y^2 , \tag{A2}$$

$$s_x^6 + s_y^6 = -\lambda^6 + 3\lambda^2 s_x^2 s_y^2 , \quad (s_x + s_y)^6 + (s_x - s_y)^6 = -2\lambda^6 - 24\lambda^2 s_x^2 s_y^2 . \tag{A3}$$

Applying a Taylor series expansion to $L_{h,+}(e^{s_x h}, e^{s_y h})$ and $L_{h,\times}(e^{s_x h}, e^{s_y h})$ and utilizing equalities (A2) and (A3), we find that

$$L_{d,+}(e^{s_x h}, e^{s_y h}) = h^2\left(\frac{1}{24}\lambda^4 - \frac{1}{6}s_x^2 s_y^2\right) + h^4\left(\frac{-1}{480}\lambda^6 + \frac{1}{120}\lambda^2 s_x^2 s_y^2\right) + O(h^6) , \tag{A4}$$

$$L_{d,\times}(e^{s_x h}, e^{s_y h}) = \frac{1}{3}h^2 s_x^2 s_y^2 - \frac{1}{30}h^4 \lambda^2 s_x^2 s_y^2 + O(h^6) . \tag{A5}$$

Since $\gamma_+ = L_{h,+}(e^{i|\lambda|h}, 1)$ and $\gamma_\times = L_{h,\times}(e^{i\frac{|\lambda|}{\sqrt{2}}h}, e^{i\frac{|\lambda|}{\sqrt{2}}h})$, we also have

$$\gamma_+ = \frac{1}{24}h^2\lambda^4 - \frac{1}{480}h^4\lambda^6 + O(h^6) , \quad \gamma_\times = \frac{1}{12}h^2\lambda^4 - \frac{1}{120}h^4\lambda^6 + O(h^6) . \tag{A6}$$

Combining (A4)-(A6), we have $\gamma_\times + \gamma_+ = O(h^2)$ and

$$(\gamma_\times + \gamma_+)L_{h,9}(e^{s_x h}, e^{s_y h}) = \gamma_\times L_{h,+}(e^{s_x h}, e^{s_y h}) + \gamma_+ L_{h,\times}(e^{s_x h}, e^{s_y h}) - \gamma_\times \gamma_+ = O(h^8) . \tag{A7}$$

We know that both $L_{h,+}$ and $L_{h,\times}$ have an accuracy of $O(h^2)$ from (A4) and (A5), and that $L_{h,9}$ has an accuracy of $O(h^6)$ from (A7).

2.5 Numerical Examples

We use the 1D and 2D convection-diffusion equations as test problems to demonstrate the efficiency of the mode-dependent finite-difference method.

2.5.1 1D Test Problem

Consider the 1D convection-diffusion equation on [0,1]

$$\frac{d^2u}{dx^2} - a(x)\frac{du}{dx} = 0\,, \quad \text{where} \quad a(x) = 10+\epsilon x + \frac{\epsilon}{10+\epsilon x}\,, \tag{1}$$

with given $u(0)$ and $u(1)$. Our goal is to study the effect of the linear perturbation term ϵx on the accuracy of the mode dependent discretization scheme described in Section 3. Note that when $\epsilon = 0$, the coefficient $a(x)$ is constant, and according to our analysis we expect that in this case the mode-dependent discretization will be exact. The term $\frac{\epsilon}{10+\epsilon x}$ is added so that (1) has the following analytic solution

$$u(x) = u(0) + [\,u(1) - u(0)\,]\,\frac{\exp(10x + 0.5\epsilon x^2) - 1}{\exp(10+0.5\epsilon) - 1}\,, \quad \text{where } \exp(x) = e^x\,.$$

The boundary conditions $u(0) = 1$ and $u(1) = 10$ are selected. We compare the conventional and mode-dependent central difference schemes, i.e.

$$(1-\frac{a_n h}{2})u_{n+1} - 2u_n + (1+\frac{a_n h}{2})u_{n-1} = 0\,, \tag{2a}$$

$$\exp(-\frac{a_n h}{2})u_{n+1} - 2\cosh(\frac{a_n h}{2})u_n + \exp(\frac{a_n h}{2})u_{n-1} = 0\,, \tag{2b}$$

where $h = \frac{1}{N}$, $u_n = u_h(nh) \approx u(nh)$ for $1 \leqslant n \leqslant N-1$, $u_0 = u(0)$ and $u_N = u(1)$.

First, we study the effect of the grid size h when the parameter $\epsilon = 1$. Figure 2.4 shows that the errors of both schemes are proportional to $O(h^2)$. Next, we study the impact on the error of variations of the coefficient function $a(x)$. The first derivative of the coefficient function $a(x)$ is approximately measured by the parameter ϵ, so that

ϵ can be used as a measure of the local variations of $a(x)$. Errors versus ϵ for a fixed grid size $h = \frac{1}{16}$ are plotted in Figure 2.5. From this figure, we see that the conventional scheme is insensitive to changes in ϵ while the error of the mode-dependent scheme is proportional to $O(\epsilon)$.

2.5.2 2D Test Problem

The 2D test problem is the equation on $[0,1]^2$

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - \left(8+\epsilon_x x + \frac{\epsilon_x}{8+\epsilon_x x} \right) \frac{\partial u}{\partial x} - \left(6+\epsilon_y y + \frac{\epsilon_y}{6+\epsilon_y y} \right) \frac{\partial u}{\partial y} = 0 , \qquad (3)$$

with Dirichlet boundary conditions associated with the following three exact solutions

$$(a)\ \epsilon_x = \epsilon_y = 0,\ u(x,y) = \exp[(4+5\cos(\tfrac{7\pi}{8}))\, x + (3+5\sin(\tfrac{7\pi}{8}))\, y]. \qquad (4a)$$

$$(b)\ \epsilon_x = \epsilon_y = 0,\ u(x,y) = (0.2 + 6e^{8x})(0.01 + 2e^{6y}), \qquad (4b)$$

$$(c)\ \epsilon_x = \epsilon_y = 0.002,\ u(x,y) = [0.2 + 6\exp(8x + 10^{-3}x^2)][0.01 + 2\exp(6y + 10^{-3}y^2)]. \qquad (4c)$$

We use the finite-difference schemes (4.14)-(4.18) discussed in Section 4.3 to discretize (3) with grid size $h = \frac{1}{4}, \frac{1}{8}, \frac{1}{16}$ and $\frac{1}{32}$. The resulting systems of equations are solved by the SOR method for test cases (a) and (b) and by a local relaxation method described in [8], [26] and [27] for test case (c). We plot the errors versus the grid size in Figures 2.6 - 2.8.

For test case (a), the solution contains a single mode. All 5-point stencil discretization schemes have an accuracy of $O(h^2)$. The 9-point stencil discretization has an accuracy close to $O(h^6)$ when the grid sizes are still coarse.

For test case (b), the solution contains four modes 1, e^{8x}, e^{6y} and e^{8x+6y}. In this case, since the Allen-Southwell scheme catches all these modes, it should be an exact method. Thus, its error represents the numerical rounding error instead of the discretization error. The other 5-point stencil discretizations give an error proportional to $O(h^2)$ for fine grids. The 9-point stencil scheme is considerably more accurate than the other 5-point stencil schemes. It comes close to the exact method when the grid size is $\frac{1}{32}$.

Test case (c) can be viewed as obtained from test case (2) by introducing linear perturbation terms $\epsilon_x x$ and $\epsilon_y y$ with $\epsilon_x = \epsilon_y = 0.002$ in the coefficient functions. We consider the effect of small variations of the coefficient functions. The Allen-Southwell scheme is not exact any longer, but still has a high accuracy. The 9-point discretization scheme has almost the same performance as the Allen-Southwell scheme. However, if we compare Figures 2.7 and 2.8, we see that the coefficient variations due to ϵ_x and ϵ_y make the error of the 9-point scheme 10 times larger for the unperturbed case depicted in Figure 2.7. The accuracy of the other 5-point stencil schemes remains approximately the same.

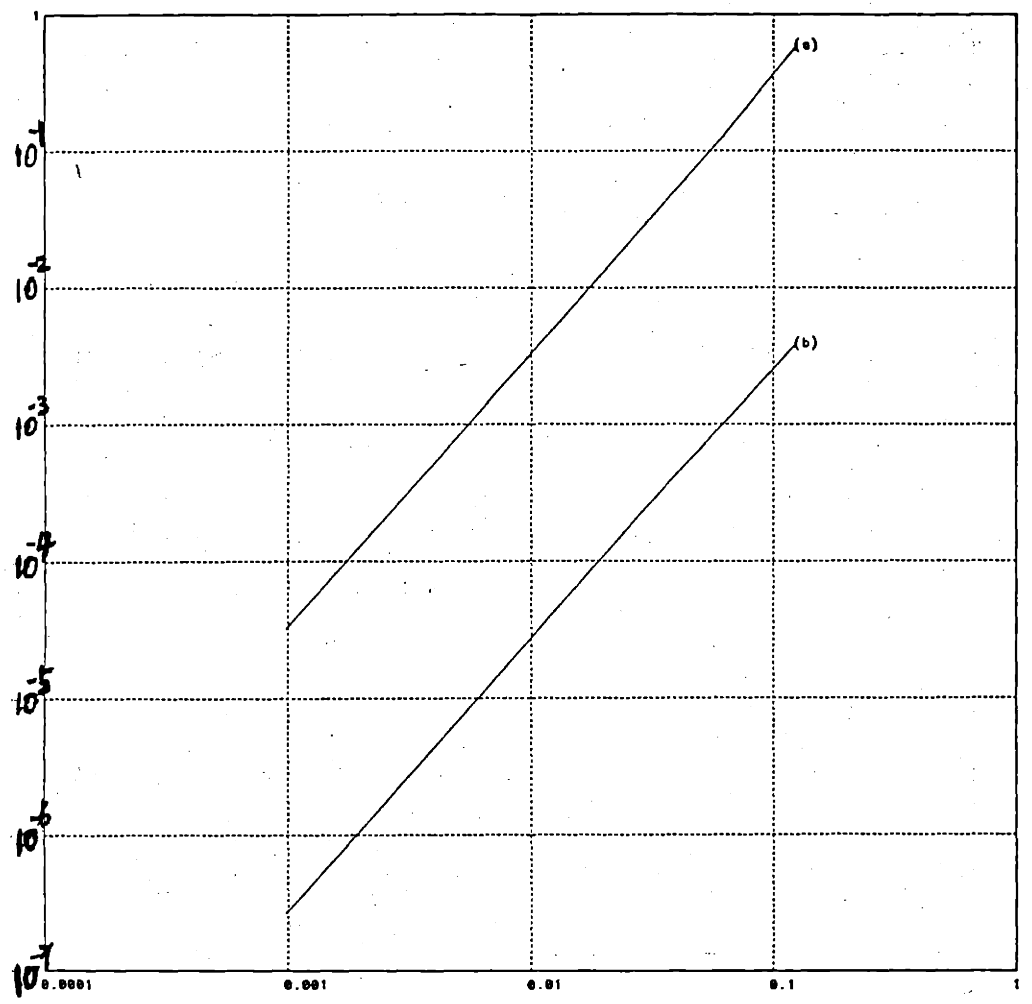


Figure 2.4: l_∞-norm of the error versus the grid size h for (1) with $\epsilon = 1$: (a) the central difference scheme and (b) the mode-dependent scheme.

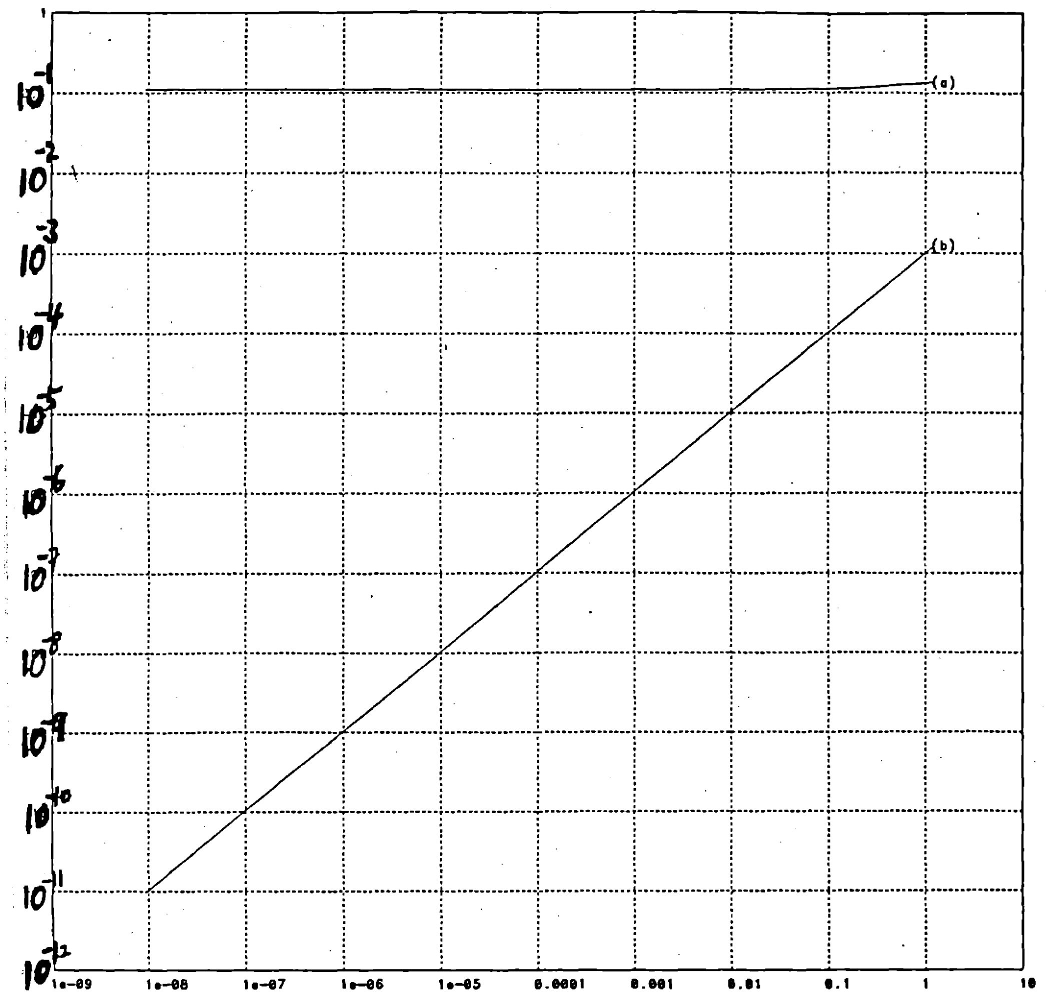


Figure 2.5: l_∞-norm of the error versus the parameter ϵ for (1) with $h = \frac{1}{16}$: (a) the central difference scheme and (b) the mode-dependent scheme.

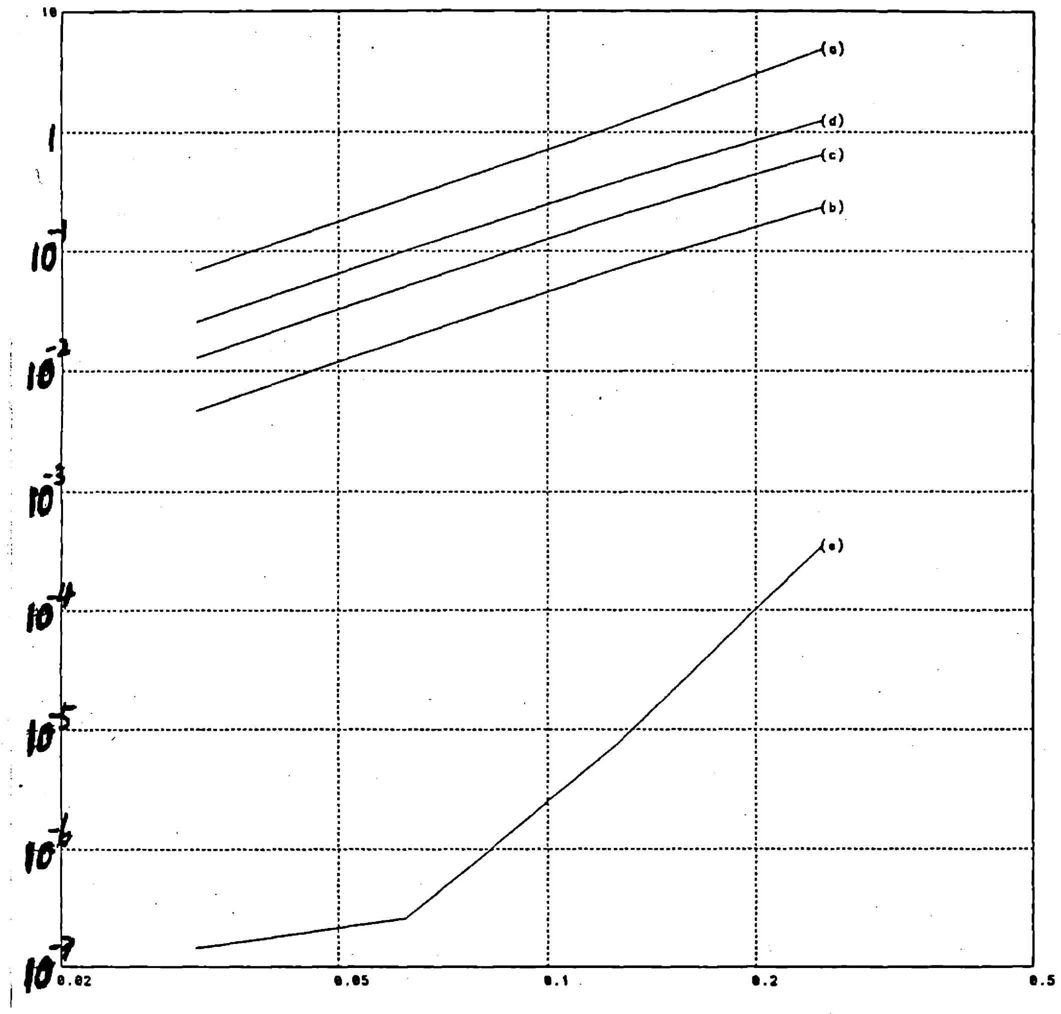


Figure 2.6: l_∞-norm of the error versus the grid size h for (4a) : (a) $L_{h,c}$, (b) $L_{h,AS}$, (c) $L_{h,+}$, (d) $L_{h,\times}$ and (e) $L_{d,9}$ given by (2.4.14)-(2.4.18).

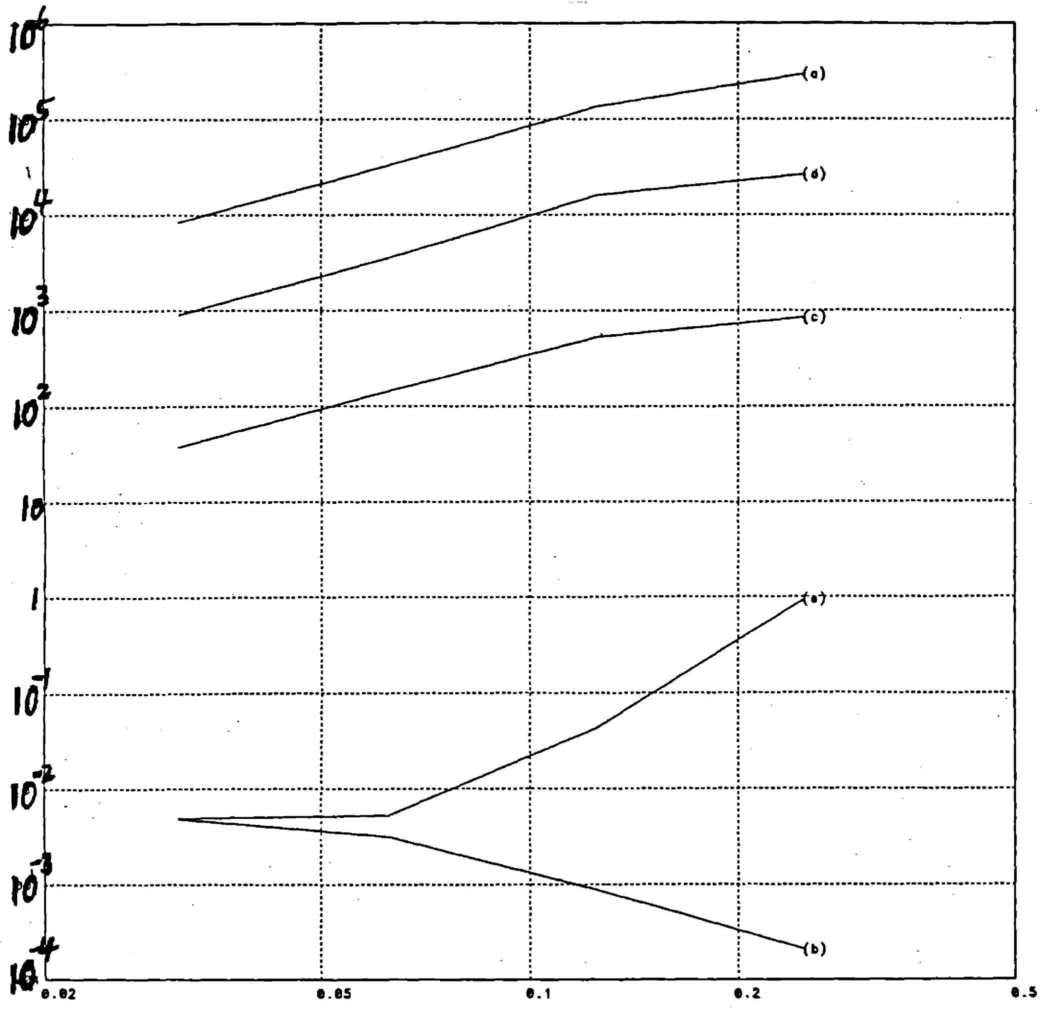


Figure 2.7: l_∞-norm of the error versus the grid size h for (4b) : (a) $L_{h,c}$, (b) $L_{h,AS}$, (c) $L_{h,+}$, (d) $L_{h,\times}$ and (e) $L_{d,9}$ given by (2.4.14)-(2.4.18).

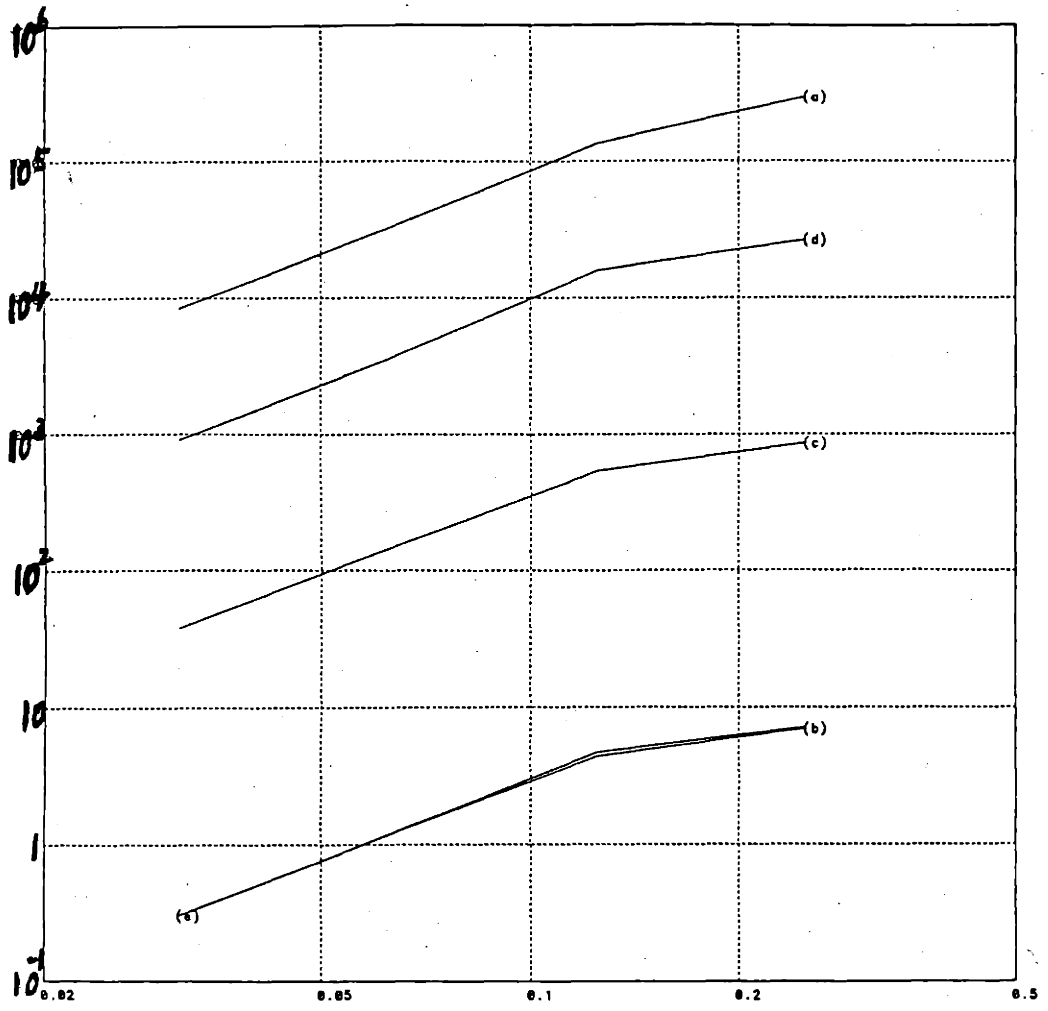


Figure 2.8: l_∞-norm of the error versus the grid size h for (4c) : (a) $L_{h,c}$, (b) $L_{h,AS}$, (c) $L_{h,+}$, (d) $L_{h,\times}$ and (e) $L_{d,9}$ given by (2.4.14)-(2.4.18).

2.6 Conclusions and Extensions

Although its properties were not always well understood, the mode-dependent finite-difference method has been discovered and rediscovered several times by a number of researchers and has been applied to the discretization of several types of ODEs and PDEs.

As was mentioned earlier, when the cell Reynolds number is large, the conventional central difference discretization of the convection-diffusion equation has convergence difficulties. Hence, the need for a mode-dependent scheme arises naturally when discretizing this equation, and more generally, when considering singular perturbation problems. Allen and Southwell [1] presented the first discretization of this type. A more detailed investigation of this scheme was performed by Dennis [12]. Since then, there have been a number of rediscoveries and elaborations such as [3] [9] [18] [28] [29] [35] [36] [39] [40]. Applications of Allen-Southwell's scheme to 2-D or 3-D fluid flow problems can be found in [2] [13]-[17] [37] [38] [41]. Some researchers extended the mode-dependent idea to the design of finite-element methods, see e.g. [6] [10] [21]-[24] [33]. The methodology described in this chapter can also be applied to the discretization of initial value ODEs. A mode-dependent finite-difference scheme for initial-value ODEs was first studied by Gautschi [19]. Some generalizations of Gautschi's work can be found in [5] [30] [34] [42] [43].

Interestingly, the mode-dependent scheme has been introduced under a number of different names such as the locally exact technique [3], the weighted-mean scheme [18], the smart upwind method [20], the optimal finite analytic method [32] and the upstream-weighted difference scheme [35].

In this chapter, we have used the spectral structure of differential operators to obtain more accurate finite-difference schemes. The transform domain point of view was shown to be simple and useful. For the case of homogeneous ODEs, we proposed a universal mode-dependent finite-difference scheme which is exact for constant-coefficient equations, and has a very high accuracy for equations with smoothly varying coefficients. For homogeneous PDEs, we considered mode-dependent 5-point, rotated 5-point and 9-point stencil discretizations of the Laplace, Helmholtz and convection-diffusion equations. The mode-dependent schemes for the Helmholtz and convection-diffusion equations turn out to be natural extensions of the schemes derived for the Laplace equation.

There exist similarities and differences between the mode-dependent finite-difference method and spectral methods. Both discretization techniques are based on a spectral analysis of the differential and difference operators and try to match their spectral properties. However, the spectral method analyzes spectra by using Fourier basis functions, i.e. functions with frequencies along the imaginary axis. In this approach, a large number of basis functions is usually required to synthesize a given function. Hence, in order to get a high degree of accuracy, more grid points are necessary and the resulting scheme is a global one. The mode-dependent finite-difference method enlarges the set of basis functions so that the spectral analysis can be performed in the entire transform domain. Since fewer basis functions are required to synthesize a function due to this enlargement, the resulting scheme is local. This local nature of the mode-dependent finite-difference method makes it easy to analyze and insensitive to boundary conditions. In contrast, spectral methods are relatively more complicated and sensitive to different types of boundary conditions.

We basically focused on the discretization of a differential operator in the interior region and assumed the simplest Dirichlet boundary conditions throughout this chapter. Since the finite-difference method is local, the discretization scheme for grid points in the interior region will not be affected by the specific nature of the boundary conditions. However, grid points along the boundary need some special treatment. Although the general mode-dependent concept should still apply in this case, some details need to be examined in later work. In addition, as mentioned above, it would be of interest to find a general procedure for estimating the error of mode-dependent finite-difference schemes when they are applied to varying-coefficient differential equations.

2.7 References

[1] D. N. De G. Allen and R. V. Southwell, "Relaxation Methods Applied to Determine the Motion, in Two Dimensions, of a Viscous Fluid Past a Fixed Cylinder," *Q. J. Mech. and Appl. Math.*, vol. 8, pp. 129-145, 1955.

[2] D. N. De G. Allen, "A Suggested Approach to Finite-difference Representation of Differential Equations, with an Application to Determine Temperature-distributions near a Sliding Contact," *Q. J. Mech. and Appl. Math.*, vol. 15, pp. 11-33, 1962.

[3] K. E. Barrett, "The Numerical Solution of Singular-Perturbation Boundary-Value Problems," *Q. J. Mech. and Appl. Math.*, vol. 27, pp. 57-68, 1974.

[4] A. E. Berger, J. M. Solomon, M. Ciment, S. H. Leventhal, and B. C. Weinberg, "Generalized OCI Schemes for Boundary Layer Problems," *Math. Comp.*, vol. 35, no. 151, pp. 695-731, Jul. 1980.

[5] D. G. Bettis, "Numerical Integration of Products of Fourier and Ordinary Polynomials," *Numer. Math.*, vol. 14, pp. 421-434, 1970.

[6] W. S. Blackburn, "Letters to the Editor," *Int. J. Num. Meth. Eng.*, vol. 10, pp. 718-719, 1976.

[7] R. F. Boisvert, "Families of High Order Accurate Discretizations of Some Elliptic Problems," *SIAM J. Sci. Stat. Comput.*, vol. 2, no. 3, pp. 268-284, Sep. 1981.

[8] E. F. Botta and A. E. P. Veldman, "On Local Relaxation Methods and Their Application to Convection-Diffusion Equations," *Journal of Computational Physics*, vol. 48, pp. 127-149, 1981.

[9] D. G. Briggs, "A Finite Difference Scheme for the Incompressible Advection-diffusion Equation," *Comp. Meth. Appl. Mech. Eng.*, vol. 6, pp. 233-241, 1975.

[10] I. Christie, D. F. Griffiths, A. R. Mitchell, and O. C. Zienkiewicz, "Finite Element Methods for Second Order Differential Equations with Significant First Derivatives," *Int. J. Num. Meth. Eng.*, vol. 10, pp. 1389-1396, 1976.

[11] G. Dahlquist, A. Bjorck, and N. Anderson, *Numerical Methods*. Englewood Cliffs, N.J. : Prentice-Hall, Inc. , 1974.

[12] S. C. R. Dennis, "Finite Differences Associated with Second-Order Differential Equations," *Q. J. Mech. and Appl. Math.*, vol. 13, pp. 487-507, 1960.

[13] S. C. R. Dennis, "The Numerical Solution of the Vorticity Transport Equation," in *Proc. of Third Int. Conf. Num. Meth. Fluid Mech.*, New York, NY: Springer-Verlag, 1973, pp. 120–129.

[14] S. C. R. Dennis, D. B. Ingham, and R. N. Cook, "Finite-Difference Methods for Calculating Steady Incompressible Flows in Three Dimensions," *J. Comp. Phys.*, vol. 33, pp. 325-339, 1979.

[15] S. C. R. Dennis, D. B. Ingham, and S. N. Singh, "The Steady Flow of a Viscous Fluid Due to a Rotating Sphere," *Q. J. Mech. Appl. Math.*, vol. 34, pp. 361-381, 1981.

[16] S. C. R. Dennis, D. B. Ingham, and S. N. Singh, "The Slow Translation of a Sphere in a Rotating Viscous Fluid," *J. Fluid Mech.*, vol. 117, pp. 251-267, 1982.

[17] T. M. El-Mistikawy and M. J. Werle, "Numerical Method for Boundary Layers with Blowing - The Exponential Box Scheme," *AIAA J.*, vol. 16, pp. 749-751, Jul. 1978.

[18] M. E. Fiadeiro and G. Veronis, "On Weighted-mean Schemes for the Finite-difference Approximation to the Advection-diffusion Equation," *Tellus*, vol. 29, pp. 512-522, 1977.

[19] W. Gautschi, "Numerical Integration of Ordinary Differential Equations Based on Trigonometric Polynomials," *Numer. Math.*, vol. 3, pp. 381-397, 1961.

[20] P. M. Gresho and R. L. Lee, "Don't Suppress the Wiggles - They're Telling You Something," *Computers and Fluids*, vol. 9, pp. 223-253, 1981.

[21] J. C. Heinrich, P. S. Huyakorn, O. C. Zienkiewicz, and A. R. Mitchell, "An 'Upwind' Finite Element Scheme for Two-Dimensional Convection Transport Equation," *Int. J. Num. Meth. Eng.*, vol. 11, pp. 131-143, 1977.

[22] J. C. Heinrich and O. C. Zienkiewicz, "Quadratic Finite Element Schemes for Two-dimensional Convective-Transport Problems," *Int. J. Num. Meth. Eng.*, vol. 11, pp. 1831-1844, 1977.

[23] P. S. Huyakorn, "Solution of Steady-state, Convective Transport Equation Using an Upwind Finite Element Scheme," *Appl. Math. Modelling*, vol. 1, pp. 187-195, Mar. 1977.

[24] A. Kanarachos, "Boundary Layer Refinements in Convective Diffusion Problems," *Int. J. Num. Meth. Eng.*, vol. 18, pp. 167-180, 1982.

[25] L. V. Kantorovich and V. I. Krylov, *Approximate Methods of Higher Analysis.* New York: Interscinece Publishers, Inc., 1964.

[26] C.-C. J. Kuo, B. C. Levy, and B. R. Musicus, "A Local Relaxation Method for Solving Elliptic PDEs on Mesh-Connected Arrays," to apear in *SIAM Journal of Scientific and Statistical Computing*.

[27] C.-C. J. Kuo, B. C. Levy, "A Two-level Four-color SOR method," Report LIDS-P-1625, Laboratory for Information and Decision Systems, MIT, Cambridge, MA, Nov. 1986.

[28] B. P. Leonard, "A Survey of Finite Differences with Upwinding for Numerical Modelling of the Imcompressible Convective Diffusion Equation," in *Computational Techniques in Transient and Turbulent Flow*, ed. C. Taylor & K. Morgan, Swansea, U.K.: Pineridge Press, 1981.

[29] W. Lick and T. Gaskins, "A Consistent and Accurate Procedure for Obtaining Difference Equations from Differential Equations," *Int. J. Num. Meth. Eng.*, vol. 20, pp. 1433-1441, 1984.

[30] T. Lyche, "Chebyshevian Multistep Methods for Ordinary Differential Equations," *Numer. Math.*, vol. 19, pp. 65-75, 1972.

[31] R. E. Lynch and J. R. Rice, "A High-Order Difference Method for Differential Equations," *Mathematics of Computation*, vol. 34, no. 150, pp. 333-372, Apr. 1980.

[32] R. Manohar and J. W. Stephenson, "Optimal Finite Analytic Methods," *J. Heat Transfer*, vol. 104, pp. 432–437, Aug. 1982.

[33] A. Moult, D. Burley, and H. Rawson, "The Numerical Solution of Two-Dimensional, Steady Flow Problems by the Finite Element Method," *Int. J. Num. Meth. Eng.*, vol. 14, pp. 11-35, 1979.

[34] B. Neta and C. H. Ford, "Families of Methods for Ordinary Differential Equations Based on Trigonometric Polynomials," *Journal of Computational and Applied Mathematics*, vol. 10, pp. 33-38, 1984.

[35] G. D. Raithby and K. E. Torrance, "Upstream-Weighted Differencing Schemes and Their Application to Elliptic Problems Involving Fluid Flow," *Computers and Fluids*, vol. 2, pp. 191-206, 1974.

[36] D. F. Roscoe, "New Methods for the Derivation of Stable Difference Representations for Differential Equations," *J. Inst. Maths Applics*, vol. 16, pp. 291-301, 1975.

[37] D. F. Roscoe, "The Solution of the Three-Dimensional Navier-Stokes Equations Using a New Finite Difference Approach," *Int. J. Num. Meth. Eng.*, vol. 10, pp. 1299-1308, 1976.

[38] D. F. Roscoe, "The Numerical Solution of the Navier-Stokes Equations for a Three-Dimensional Laminar Flow in Curved Pipes Using Finite-Difference Methods," *J. Eng. Math.*, vol. 12, no. 4, pp. 303-323, Oct. 1978.

[39] A. K. Runchal, "Convergence and Accuracy of Three Finite Difference Schemes for a Two-dimensional Conduction and Convection Problem," *Int. J. Num. Meth. Eng.*, vol. 4, pp. 541-550, 1972.

[40] D. B. Spalding, "A Novel Finite Difference Formulation for Differential Expressions Involving Both First and Second Derivatives," *Int. J. Num. Meth. Eng.*, vol. 4, pp. 551-559, 1972.

[41] N. C. Steele and K. E. Barrett, "A 2nd Order Numerical Method for Laminar Flow at Moderate to High Reynolds Numbers: Entrance Flow in a Duct," *Int. J. Num. Meth. Eng.*, vol. 12, pp. 405-414, 1978.

[42] E. Stiefel and D. G. Bettis, "Stabilization of Cowell's Method," *Numer. Math.*, vol. 13, pp. 154-175, 1969.

[43] P. J. Van Der Houwen and B. P. Sommeijer, "Linear Multistep Methods with Reduced Truncation Error for Periodic Initial-value Problems," *IMA Journal of Numerical Analysis*, vol. 4, pp. 479-489, 1984.

PART II : SINGLE-GRID METHODS

The second part of this thesis examines iterative solution schemes, especially the SOR iteration, for elliptic PDEs discretized with a single uniform grid. It contains two chapters.

A local relaxation method is proposed in Chapter 4 to solve a class of elliptic PDE problems whose discretized form can be written in terms of a symmetric positive definite matrix. We prove the convergence of the local relaxation algorithm and use a Fourier analysis approach to analyze the relaxation method and to determine the optimal local relaxation parameters.

In Chapter 5, a 2-level 4-color SOR method is proposed for the 9-point discretization of the Poisson equation on a square. Instead of examining the Jacobi iteration matrix in the space domain, we consider an equivalent but much simpler 4-color iteration matrix in the frequency domain. A 2-level SOR method is introduced to increase the convergence rate for the frequency-domain iteration matrix. At a first level, the red and orange points, and then the black and green points are treated as groups, and a block SOR iteration is performed on these two groups. At a second level, another SOR iteration is used to decouple values at the red and orange points, and then at the black and green points. The conventional red/black SOR iteration for a 5-point stencil is shown to be a degenerate case of the general 2-level 4-color SOR method. For the case of the 9-point stencil, a closed-form expression for the optimal relaxation parameters ω_b^* and ω_p^* at the two iteration levels is given.

Chapter 3 : Local Relaxation Method

3.1 Introduction

The research described in this section has an objective to solve a 2-D linear elliptic PDE on a square discretized by a finite-difference method with a multiprocessor array. Suppose we assign one processor to each grid point and connect every processor to its four nearest neighbors. This kind of computer architecture, known as a mesh-connected processor array, suggests a natural parallel computation scheme to solve the above system of equations, i.e., parallel computation in the space domain.

Jacobi and Gauss-Seidel relaxation methods seem particularly suitable for mesh-connected processors, since each processor uses only the most recent values computed by its neighbors to update its own value. Unfortunately, the convergence rate of these algorithms is slow. The convergence rate can be improved by various acceleration schemes such as successive over-relaxation (SOR) and Chebyshev semi-iterative relaxation (CSI) [17]. However, to obtain the acceleration effect requires that the acceleration factors should be estimated adaptively [8]. This procedure requires global communication on a mesh-connected processor array and increases the computation time per iteration enormously. Any time savings due to acceleration may be canceled out by the increased communication time. In order to improve the convergence rate as well as to avoid global communication, a recently developed approach known as the ad hoc SOR [5] [6] or local relaxation [3] method seems to be useful.

The local relaxation scheme was found empirically by Ehrlich [5] [6] and Botta and Veldman [3]. They applied this method to a very broad class of problems and found its efficiency by studying many numerical examples. In this chapter, we approach the same problem from an analytical point of view, clearly prove the convergence of this method for the case of symmetric positive-definite matrices, and provide an analytical explanation for the good performance of the method.

The conventional way to analyze the SOR method is to use matrix analysis [17]. This approach depends heavily on the ordering of the grid points and on the properties of the resulting sparse matrix. An alternative technique, which was employed in [12], [14], and [15] to analyze relaxation algorithms, is to use Fourier analysis. Strictly speaking, Fourier analysis applies only to linear constant coefficient PDEs on an infinite domain, or with periodic boundary conditions. Nevertheless, at a heuristic level this approach provides a useful tool for the analysis of more general PDE problems, and it has been used by Brandt [4] to study the error smoothing effect of relaxation algorithms and to develop multigrid methods. Since then, the Fourier analysis approach has received a large amount of attention in the study of multigrid methods [16]. Following the same idea, we shall apply the Fourier analysis approach to the SOR method. For the Poisson Problem defined on the unit square with Dirichlet boundary conditions, we obtain the same result as Young's SOR method. However, our derivation is simpler. For space-varying PDEs, the local relaxation scheme uses space adaptive relaxation parameters. This is different from Young's SOR method which uses time adaptive relaxation parameters [8].

This chapter is organized as follows. Section 3.2 proves the convergence of the local relaxation method. The Fourier analysis approach is used to determine the optimal relaxation parameteres of the local relaxation method for 5-point and 9-point stencils respectively in Sections 3.3 and 3.4. We also show that the convergence analysis of the local relaxation method for the 9-point stencil discretization requires a slight modification of the basic convergence result of Section 3.2. Section 3.5 shows the results of a computer simulation on a test problem which indicates that the convergence rate of the local relaxation method is superior to that of the adaptive SOR method. Some further extensions and conclusions are mentioned in Section 3.6. The implementation of the Jacobi, Gauss-Seidel, adaptive SOR, and local relaxation methods on mesh-connected processor arrays will be discussed in a separate chapter, i.e. Chapter 6.

3.2 Local Relaxation Algorithm

Consider a self-adjoint second-order linear PDE defined on a closed unit square $\Omega = [0,1] \times [0,1]$,

$$-\frac{\partial}{\partial x}\{p(x,y)\frac{\partial u}{\partial x}\} - \frac{\partial}{\partial y}\{q(x,y)\frac{\partial u}{\partial y}\} + \sigma(x,y)u = f(x,y), \quad (x,y) \in \Omega, \tag{1a}$$

with the following boundary condition on Γ, the boundary of Ω,

$$\alpha(x,y)u + \beta(x,y)\frac{\partial u}{\partial n} = \gamma(x,y), \quad (x,y) \in \Gamma, \tag{1b}$$

where $\frac{\partial u}{\partial n}$ denotes the outward derivative normal to Γ. The coefficient functions are assumed to be smooth and to satisfy

$$p(x,y) > 0, \quad q(x,y) > 0, \quad \sigma(x,y) \geqslant 0, \ (x,y) \in \Omega,$$
$$\alpha(x,y) \geqslant 0, \quad \beta(x,y) \geqslant 0, \quad \alpha + \beta > 0, \ (x,y) \in \Gamma.$$

If we discretize (1) on a uniform grid Ω_h with grid spacing $h_x = h_y = 1$ by a 5-point stencil, the finite difference equation at an interior point (n_x, n_y) can be written as [17]

$$d_{n_x,n_y} u_{n_x,n_y} - r_{n_x,n_y} u_{n_x+1,n_y} - l_{n_x,n_y} u_{n_x-1,n_y} - t_{n_x,n_y} u_{n_x,n_y+1}$$
$$- b_{n_x,n_y} u_{n_x,n_y-1} = s_{n_x,n_y}, \tag{2}$$

with

$$l_{n_x,n_y} = p_{n_x-½,n_y}, \ r_{n_x,n_y} = p_{n_x+½,n_y}, \ b_{n_x,n_y} = q_{n_x,n_y-½}, \ t_{n_x,n_y} = q_{n_x,n_y+½}, \tag{3a}$$

$$d_{n_x,n_y} = p_{n_x-½,n_y} + p_{n_x+½,n_y} + q_{n_x,n_y-½} + q_{n_x,n_y+½} + \sigma_{n_x,n_y}h^2, \ s_{n_x,n_y} = f_{n_x,n_y}h^2 \tag{3b}$$

where p_{n_x,n_y} is defined as $p(n_x h, n_y h)$. Similar discretized equations can be obtained for the boundary points where u_{n_x,n_y} is unknown. Let us choose a particular order for those equations, and construct vectors u and s from the variables u_{n_x,n_y} and s_{n_x,n_y} arranged in the selected order; then the interior and boundary equations can be arranged in matrix form

$$A\ u = s\ , \tag{4}$$

where A contains the coefficients d_{n_x,n_y}, l_{n_x,n_y}, r_{n_x,n_y}, t_{n_x,n_y}, and b_{n_x,n_y}. The matrix A is symmetric, since $l_{n_x+1,n_y} = r_{n_x,n_y}$ and $b_{n_x,n_y+1} = t_{n_x,n_y}$. In addition, A is positive definite, since it is irreducibly diagonal dominant [17, p. 23].

A particularly simple iteration for the solution of (4) is the Jacobi method which can be written as

$$u_{n_x,n_y}^{m+1} = d_{n_x,n_y}^{-1}\ (l_{n_x,n_y}\ u_{n_x-1,n_y}^{m} + r_{n_x,n_y}\ u_{n_x+1,n_y}^{m} + b_{n_x,n_y}\ u_{n_x,n_y-1}^{m} + t_{n_x,n_y}\ u_{n_x,n_y+1}^{m} + s_{n_x,n_y}\ ).$$

Another simple iteration for this problem is the Gauss-Seidel relaxation with red/black point partitioning. The corresponding local equations can be written as

red points ($n_x + n_y$ is even) :

$$u_{n_x,n_y}^{m+1} = d_{n_x,n_y}^{-1}\ (l_{n_x,n_y}\ u_{n_x-1,n_y}^{m} + r_{n_x,n_y}\ u_{n_x+1,n_y}^{m} + b_{n_x,n_y}\ u_{n_x,n_y-1}^{m} + t_{n_x,n_y}\ u_{n_x,n_y+1}^{m} + s_{n_x,n_y}\ ),$$

black points ($n_x + n_y$ is odd) :

$$u_{n_x,n_y}^{m+1} = d_{n_x,n_y}^{-1}\ (l_{n_x,n_y}\ u_{n_x-1,n_y}^{m+1} + r_{n_x,n_y}\ u_{n_x+1,n_y}^{m+1} + b_{n_x,n_y}\ u_{n_x,n_y-1}^{m+1} + t_{n_x,n_y}\ u_{n_x,n_y+1}^{m+1} + s_{n_x,n_y}\ ).$$

Note that the difference between the Jacobi and the Gauss-Seidel relaxation methods is that the Jacobi method updates the values of all nodes at one iteration while the Gauss-Seidel method updates the values of half of these nodes during a first step and updates the values of the other half during a second step based on the previously updated information, and these two steps form a complete iteration. The chief shortcoming of the Jacobi or Gauss-Seidel iterative methods lies in their slow convergence rate. The spectral radius of the relaxation matrix is equal to $1 - O(h^2)$. Therefore, in order to let the difference of the computed and exact solutions be within the accuracy of the discretization error $O(h^2)$ of (2), the number of iterations needed is proportional to $O(h^{-2})$ [17].

By applying different acceleration schemes to the Jacobi and Gauss-Seidel techniques, we can derive a variety of accelerated relaxation algorithms. Two typical examples are the Chebyshev semi-iterative (CSI) method and the successive over-relaxation (SOR) method. These acceleration schemes use carefully chosen relaxation parameters to reduce the spectral radii of the iterative matrices so that the iterative algorithms converge faster. To determine the relaxation parameters, CSI acceleration uses knowledge of the largest and smallest eigenvalues of the basic relaxation matrix and SOR acceleration uses knowledge of the spectral radius of the basic relaxation matrix [17]. To our knowledge, all the estimation procedures developed require the computation of the norms of some global vectors.

The local relaxation method proposed by Ehrlich [5] [6] and Botta and Veldman [3] can be written as

red points ($n_x + n_y$ is even) :

$$u^{m+1}_{n_x,n_y} = (1-\omega_{n_x,n_y})\, u^m_{n_x,n_y} \tag{5a}$$

$$+\,\omega_{n_x,n_y}\, d^{-1}_{n_x,n_y}\, (l_{n_x,n_y}\, u^m_{n_x-1,n_y} + r_{n_x,n_y}\, u^m_{n_x+1,n_y} + b_{n_x,n_y}\, u^m_{n_x,n_y-1} + t_{n_x,n_y}\, u^m_{n_x,n_y+1} + s_{n_x,n_y}),$$

black points ($n_x + n_y$ is odd) :

$$u^{m+1}_{n_x,n_y} = (1-\omega_{n_x,n_y})\, u^m_{n_x,n_y} \tag{5b}$$

$$+\,\omega_{n_x,n_y}\, d^{-1}_{n_x,n_y}\, (l_{n_x,n_y}\, u^{m+1}_{n_x-1,n_y} + r_{n_x,n_y}\, u^{m+1}_{n_x+1,n_y} + b_{n_x,n_y}\, u^{m+1}_{n_x,n_y-1} + t_{n_x,n_y}\, u^{m+1}_{n_x,n_y+1} + s_{n_x,n_y}),$$

where ω_{n_x,n_y} is called the *local relaxation* parameter.

Assuming Dirichlet boundary conditions and $N_x \times N_y = N$ unknowns within the unit square, it was suggested in [5] [6] that a good choice of local relaxation parameters ω_{n_x,n_y} is given by

$$\omega_{n_x,n_y} = \frac{2}{1+(1-\rho^2_{n_x,n_y})^{1/2}}, \tag{6}$$

where

$$\rho_{n_x,n_y} = \frac{2}{d_{n_x,n_y}}[(l_{n_x,n_y}\, r_{n_x,n_y})^{1/2}\cos\frac{\pi}{N_x+1} + (t_{n_x,n_y}\, b_{n_x,n_y})^{1/2}\cos\frac{\pi}{N_y+1}]. \tag{7}$$

Since we consider only the case of symmetric discretized matrices, the parameter ρ_{n_x,n_y} is always real. This gives us the ad hoc SOR method or the local relaxation method for a symmetric matrix. However, the local relaxation method can also be applied to more general matrices such that ρ_{n_x,n_y} is purely imaginary or complex. An ad hoc formula to determine the local relaxation parameters for these cases can be found in [3],[5] and [6]. In this chapter, we will focus on the local relaxation method for a system of equations $A\ u = s$ where A is symmetric positive definite.

Although the local relaxation method was empirically shown to be powerful, there are several questions which were left unanswered by the papers of Ehrlich, Botta, and Veldman. First, they did not prove that the local relaxation method converges. Furthermore, there was no explanation of why the local relaxation method converges very fast. In the following sections, we will explore these two issues.

3.3 Convergence Analysis

In this section, we give a sufficient condition for the convergence of a local relaxation procedure. Then, we show that the local relaxation method given by equations (3.2.5) - (3.2.7) indeed converges.

In order to obtain a convergence result which covers the most general type of local relaxation procedure, we use a matrix formulation, since such a formulation includes not only the 5-point stencil corresponding to the discretized equation (3.2.2), but also other kinds of stencils. Given a linear system of equations, $A\ u = s$, where A is an $N \times N$ real symmetric positive definite matrix with positive diagonal elements, we may rewrite A as

$$A = D - E - F = D\,(I - L - U),\ \text{and}\quad E^T = F,$$

where I, D, E and F represent identity, diagonal, lower and upper triangular matrices, and $L = D^{-1}E$ and $U = D^{-1}F$. Let W be the diagonal matrix formed by the local relaxation parameters, i.e., $W = \text{diag}\,(\omega_1, \omega_2, \ldots, \omega_N)$. Then, a local relaxation procedure can be written in matrix iterative form as

$$u^{m+1} = (I - W\,L)^{-1}[(I - W) + W\,U]u^m + (I - W\,L)^{-1}W\,D^{-1}s\,. \qquad (1)$$

Let $\bar{u}$ be the solution of the above iterative equation, so that

$$\bar{u} = (I - W\,L)^{-1}[(I - W) + W\,U]\bar{u} + (I - W\,L)^{-1}W\,D^{-1}s\,.$$

Define the error vector at n th iteration as $e^{(m)} = u^{(m)} - \bar{u}$. Then the matrix iterative equation in the error space becomes

$$e^{m+1} = (I - W\,L)^{-1}[(I - W) + W\,U]e^m\,. \qquad (2)$$

The iteration matrix of the local relaxation procedure (1) is therefore given by $G_W = (I - W\,L)^{-1}[(I - W) + W\,U]$. The iteration procedure will converge for all initial estimates $\bar{u}^{(0)}$ if and only if all eigenvalues of G_W are less than one in magnitude, i.e. if the spectral radius $\rho[G_W]$ of the iteration matrix G_W is less than 1. A simple sufficient condition for convergence is given by the following theorem.

Theorem 3.1 (Sufficient Condition for the Convergence of a Local Relaxation Procedure)
Suppose A is an $N \times N$ real symmetric positive definite matrix. For the local relaxation procedure given by equation (1), if $0 < \omega_n < 2$ for $1 \leqslant n \leqslant N$, then $\rho[G_W] < 1$ and the iterative algorithm converges.

Proof:

Let λ and p be an arbitrary eigenvalue, eigenvector pair of G_W. Then $G_W\, p = \lambda\, p$, or equivalently,

$$[(I - W) + W\, U]\, p = \lambda (I - W\, L)\, p \,. \tag{3}$$

Premultiplying by $p^H\, D\, W^{-1}$ on both sides, we obtain

$$p^H D\, W^{-1} p - p^H D\, p + p^H D\, U\, p = \lambda\, p^H D\, W^{-1} p - \lambda\, p^H D\, L\, p \,.$$

Since $E^T = F$, $E = D\, L$, and $F = D\, U$, it is easy to check that

$$p^H D\, U\, p = (L\, p)^H D\, p = \overline{p^H D\, L\, p}$$

Defining $z = \dfrac{p^H D\, L\, p}{p^H D\, p}$ and $\dfrac{1}{\omega} = \dfrac{p^H D\, W^{-1} p}{p^H D\, p}$ the equation (3) can be simplified as

$$\frac{1}{\omega} - 1 + \bar{z} = \frac{\lambda}{\omega} - \lambda z \ ,$$

or equivalently,

$$\lambda = \frac{1 - \omega + \omega \bar{z}}{1 - \omega z} \,.$$

Let $z = r\, e^{i\theta}$, then

$$|\lambda|^2 = \lambda \bar{\lambda} = 1 - \frac{\omega(2-\omega)(1 - 2r\cos\theta)}{(1 - \omega r \cos\theta)^2 + \omega^2 r^2 \sin^2\theta} \tag{4}$$

We know that $| \lambda |^2$ is always positive. If we can show that the second term in the above expression is also positive, then we can conclude that $| \lambda |$ is less than 1. The denominator of the second term of equation (4) is positive, so that we only have to consider the numerator. We have

$$2 r \cos\theta = 2 \operatorname{Re}(z) = \bar{z} + z = \frac{p^H D L p}{p^H D p} + \frac{p^H D U p}{p^H D p}$$

$$= 1 - \frac{p^H A p}{p^H D p} < 1 ,$$

where the inequality is due to the fact that A and D are both positive definite. Note that since A is positive definite, the matrix D formed with the diagonal elements of A is also positive definite. Therefore, we know that $1 - 2 r \cos\theta > 0$. Now, consider the range of the parameter ω. Since $W = \operatorname{diag}(\omega_1 , \omega_2 , \dots , \omega_N)$,

$$W^{-1} = \operatorname{diag}(\omega_1^{-1} , \omega_2^{-1} , \dots , \omega_N^{-1}) .$$

Assuming that all relaxation factors are positive, we have

$$\frac{1}{\omega_{\max}} < \frac{p^H D W^{-1} p}{p^H D p} = \frac{\sum_{n=1}^{N} | p_n |^2 d_n \omega_n^{-1}}{\sum_{n=1}^{N} | p_n |^2 d_n} < \frac{1}{\omega_{\min}} ,$$

where $\omega_{\max}$ and $\omega_{\min}$ are the largest and smallest eigenvalues of the matrix W and p_n is the n th element of the vector p. If we set $0 < \omega_{\min} \leqslant \omega_{\max} < 2$, then

$$0 < \omega_{\min} \leqslant \omega \leqslant \omega_{\max} < 2 .$$

Under this condition, the second term in equation (4) is always positive, so that the eigenvalues of the matrix G_W are all less than 1 and the local relaxation procedure (1) converges.

Q.E.D.

The above theorem gives the range of the local relaxation parameters which guarantees that a local relaxation procedure converges; however, it does not tell us how to choose the relaxation parameters to make a local relaxation procedure converge faster. The local relaxation method mentioned in the last section is a special case of a local relaxation procedure, where the local relaxation parameters are specified for a 5-point stencil discretization. To show its convergence, we only have to show that all relaxation parameters chosen by the rule (3.2.6), (3.2.7) are between 0 and 2.

Corollary (Convergence of the Local Relaxation Method for a 5-point Stencil Discretization)
The local relaxation method for a 5-point stencil given by (3.2.5) - (3.2.7) converges

---

Proof:

From the discussion in the previous section, we know that the matrix A obtained by discretizing (1) is symmetric positive definite.

Since $p(x,y)$ and $q(x,y)$ are positive functions and $\sigma(x,y)$ is a nonnegative function, we know from (3.2.3) that l_{n_x,n_y}, r_{n_x,n_y}, b_{n_x,n_y}, t_{n_x,n_y}, and d_{n_x,n_y} are all positive. In addition, $0 < \cos\frac{\pi}{P+1} < 1$ for $1 < P < \infty$. Therefore ρ_{n_x,n_y} given by (3.2.7) is also positive. Using the inequalities

$$2\,(l_{n_x,n_y} r_{n_x,n_y})^{1/2} \leq l_{n_x,n_y} + r_{n_x,n_y} \;, \qquad 2\,(t_{n_x,n_y} b_{n_x,n_y})^{1/2} \leq t_{n_x,n_y} + b_{n_x,n_y} \;,$$

we have

$$\rho_{n_x,n_y} = \frac{2}{d_{n_x,n_y}}\left[\, (l_{n_x,n_y} r_{n_x,n_y})^{1/2} \cos\frac{\pi}{N_x+1} + (t_{n_x,n_y} b_{n_x,n_y})^{1/2} \cos\frac{\pi}{N_y+1} \,\right]$$

$$\leq \frac{l_{n_x,n_y} + r_{n_x,n_y}}{d_{n_x,n_y}} \cos\frac{\pi}{N_x+1} + \frac{t_{n_x,n_y} + b_{n_x,n_y}}{d_{n_x,n_y}} \cos\frac{\pi}{N_y+1}$$

$$< \frac{l_{n_x,n_y} + r_{n_x,n_y} + t_{n_x,n_y} + b_{n_x,n_y}}{d_{n_x,n_y}} \leqslant 1$$

where the last inequality is obtained by noting that $\sigma_{n_x,n_y} \geqslant 0$ in equation (3b). It is easy to see that

$$0 < \omega_{n_x,n_y} = \frac{2}{1 + (1 - \rho_{n_x,n_y}^2)^{1/2}} < 2$$

for $0 < \rho_{n_x,n_y} < 1$. The local relaxation parameters chosen by the local relaxation method satisfy the sufficient condition given in Theorem 3.1, so that the relaxation method converges.

Q.E.D.

3.4 Determination of Local Relaxation Parameters

The convergence rate of a local relaxation procedure depends on how we choose the local relaxation parameters. The conventional SOR method chooses a spatially invariant relaxation parameter $\omega_{n_x,n_y} = \omega$ to minimize the asymptotic convergence rate, or, equivalently, minimize the spectral radius of G_W. Young [18] showed that the optimal choice for ω in the accelerated Gauss-Seidel iteration is

$$\omega = \frac{2}{1 + (1 - \rho^2)^{1/2}}$$

where ρ is the spectral radius of $D^{-1}(E+F)$. For this relaxation parameter, all eigenvalues of G_W, where $W = \omega I$, can be shown to have magnitude $\omega - 1$. In practice, it is quite difficult to calculate ρ exactly, and thus adaptive procedures are required to estimate ρ as the computation proceeds. In this section, we will use a Fourier analysis approach to derive a simple formula for a spatially varying relaxation parameter. Our formula is identical to that suggested by Ehrlich [5]. Our approach demonstrates that this formula will indeed achieve an excellent convergence rate. This study also gives some new insight into Young's SOR method.

For a linear constant coefficient PDE with Dirichlet or periodic boundary conditions, the eigenfunctions of $D^{-1}(E+F)$ are sinusoidal functions. Therefore, the spectral radius of this iterative matrix can be obtained by using Fourier analysis. However, for a space-varying coefficient PDE with general boundary conditions, the sinusoidal functions are not eigenfunctions. As a consequence, Fourier analysis cannot be applied rigorously. Notwithstanding this disadvantage, Fourier analysis is still a convenient tool for understanding the convergence properties of relaxation methods [16]. A more rigorous treatment to make Fourier analysis applicable to space-varying coefficient PDEs with general boundary conditions is needed and is currently under study. Roughly speaking, the reason why Fourier analysis often works in spatially varying PDE problems is that the eigenfunctions can be regarded as sinusoidal functions plus some perturbations. As long as the perturbation is comparatively small, the sinusoidal function is a good approximation of the original eigenfunction. Therefore, Fourier analysis is still a good analytical tool. A detailed formulation of Fourier analysis in this general context will be presented elsewhere.

In Section 3.1, we will show how to find the lowest Fourier component for given boundary conditions. Then, we use Fourier analysis to analyze the Jacobi relaxation method in Section 3.2. This approach is sometimes called the *local Fourier analysis* [16]. Finally, we justify the efficiency of the local relaxation method. The derivation can be viewed as a generalization of Brandt's local Fourier analysis to the Successive Over-Relaxation case.

3.4.1 Admissible Error Function Space and Its Lowest Fourier Component

Let Γ_i, $1 \leqslant i \leqslant 4$ denote the four boundaries of the unit square. Consider a set of linear first-order boundary conditions such as (3.2.1b) on the boundaries of the unit square,

$$B_i \, u = g_i \qquad \text{on } \Gamma_i \qquad 1 \leqslant i \leqslant 4 \, , \tag{1}$$

where B_i represents the boundary condition operator on the i-th boundary. It is more convenient to analyze the relaxation in the error space rather than in the solution space, because the error equations are homogeneous. The error formulation for the boundary conditions can be obtained as follows. Let $\bar{u}$ be the actual solution so that

$$B_i \, \bar{u} = g_i \qquad \text{on } \Gamma_i \quad 1 \leqslant i \leqslant 4 , \tag{2}$$

Subtracting (2) from (1), we obtain the homogeneous PDE in the error,

$$B_i \, e = 0 \qquad \text{on } \Gamma_i \quad 1 \leqslant i \leqslant 4 , \tag{3}$$

The functions defined on the unit square and satisfying the homogeneous boundary conditions (3) are called the *admissible error functions*, since any error function allowed in the relaxation process should always satisfy the given boundary conditions. All admissible error functions form the *admissible error function space*. The sinusoidal functions in the admissible error function space can be chosen as a basis of this space because of their completeness. As far as the convergence rate is concerned, we will see that only the lowest frequency component is relevant. Thus, we will find that only the lowest frequency of this basis needs to be determined.

We assume that all B_i's are constant-coefficient operators. Under this assumption, B_1 and B_3 are independent of the y-direction, B_2 and B_4 are independent of the x-direction, and since the problem domain is square, the admissible Fourier components can be written in separable form as $v_x(x)\, v_y(y)$, where $v_x(\cdot)$ and $v_y(\cdot)$ are two 1-D sinusoidal functions. The boundary condition on Γ_1 becomes

$$B_1 \, v_x(x)\, v_y(y) = v_y(y)\, B_1 \, v_x(x) = 0 \quad ,$$

i. e.,

$$B_1 \, v_x(x) = 0 \quad .$$

Similarly, we simplify the boundary conditions on Γ_2, Γ_3, and Γ_4, and decompose the 2-D problem into two independent 1-D problems.

$$\text{(I) } B_1 \, v_x(x) = 0 \quad \text{when } x = 0 , \quad B_3 \, v_x(x) = 0 \quad \text{when } x = 1 , \tag{4a}$$

(II) $B_2 v_y(y) = 0$ when $y = 0$, $B_4 v_y(y) = 0$ when $y = 1$. (4b)

From (4a) and (4b), we can determine the lowest frequencies $\hat{\kappa}_x$ and $\hat{\kappa}_y$ separately. We only show how to get $\hat{\kappa}_x$ from (I); then $\hat{\kappa}_y$ can be obtained from (II) in the same way.

Consider the mixed type boundary operators,

$$B_1 = b_1 + b_2 \frac{d}{d\,x} \qquad \text{for } x = 0\,, \tag{5a}$$

$$B_3 = b_3 + b_4 \frac{d}{d\,x} \qquad \text{for } x = 1\ . \tag{5b}$$

The Fourier component $v(\kappa_x, x)$ of $v_x(x)$ at the frequency κ_x can be written as a linear combination of two complex sinusoids $e^{i\,\kappa_x\,x}$ and $e^{-i\,\kappa_x\,x}$, i. e.,

$$s\,(\kappa_x, x\,) = c\,(\kappa_x\,)e^{i\,\kappa_x\,x} + c\,(-\kappa_x\,)e^{-i\,\kappa_x\,x} \quad . \tag{6}$$

Substituting (6) into (5), we obtain

$$(b_1 + i\ b_2\,\kappa_x\ )c\ (\kappa_x\ ) + (b_1 - i\ b_2\,k_x\ )c\ (-\kappa_x\ ) = 0$$
$$(b_3 + i\ b_4\,\kappa_x\ )e^{i\,\kappa_x\,x}c\ (\kappa_x\ ) + (b_3 - i\ b_4\,\kappa_x\ )e^{-i\,\kappa_x\,x}\ c\ (-\kappa_x\ ) = 0$$

In order to get nonzero values for $c\ (\kappa_x\ )$ and $c\ (-k_x\ )$, the determinant of the 2×2 coefficient matrix should equal zero, or equivalently

$$e^{\,i\,2\kappa_x} = \frac{(\,b_1 + i\ b_2\,\kappa_x\ )(\,b_3 - i\ b_4\,\kappa_x\ )}{(\,b_1 - i\ b_2\,\kappa_x\ )(\,b_3 + i\ b_4\,\kappa_x\ )} \tag{7}$$

Therefore, we conclude that the frequency κ_x of any admissible 1-D sinusoidal function with respect to the boundary conditions (5) must satisfy equation (7).

Let us look at two examples. If the boundary conditions on both Γ_x and Γ_3 are Dirichlet type boundary conditions, which means b_2 and b_4 are zeros, then (7) becomes

$$e^{\,i\,2\kappa_x} = 1 \quad or \quad \cos 2\,\kappa_x + i\ \sin 2\,\kappa_x = 1 \quad .$$

The solutions are $\kappa_x = k\ \pi$, $k = 0, \pm 1, \pm 2, \ldots$. However, it is easy to see that the zero frequency cannot be allowed. Thus, the lowest Fourier frequency $\hat{\kappa}_x$ in the admissible error space is π. If we change the boundary condition on Γ_3 to be of Neumann type, i. e., $b_3 = 0$ but $b_4 \neq 0$, then (7) becomes

$$e^{\,i\,2\kappa_x} = -1 \quad or \quad \cos 2\,\kappa_x + i\,\sin 2\,\kappa_x = -1 \quad .$$

The solutions are $\kappa_x = \frac{1}{2}\,k\ \pi$, where k is odd and the lowest frequency $\hat{\kappa}_x$ is $\frac{\pi}{2}$.

3.4.2 Local Jacobi Relaxation Operator and Its Properties

In this section, we use a Fourier analysis approach to analyze the local Jacobi operator and to determine its largest eigenvalue, or spectral radius, for given boundary conditions as previously discussed. The spectral radius of a local Jacobi operator will be used to determine the optimal local relaxation parameter of the local relaxation scheme in Section 3.4.3.

Define the x-direction (y-direction) forward-shift and backward-shift operators, E_x and E_x^{-1} (E_y and E_y^{-1}), as

$$\begin{array}{ll} E_x\, u_{n_x,n_y} = u_{n_x+1,n_y} & E_x^{-1}\, u_{n_x,n_y} = u_{n_x-1,n_y} \\ E_y\, u_{n_x,n_y} = u_{n_x,n_y+1} & E_y^{-1}\, u_{n_x,n_y} = u_{n_x,n_y-1} \end{array} .$$

Then, the 5-point discretization formula for an interior grid point can be written as

$$L_{n_x,n_y}\, u_{n_x,n_y} = s_{n_x,n_y}$$

where

$$L_{n_x,n_y} \equiv d_{n_x,n_y} - (\, r_{n_x,n_y}\, E_x + l_{n_x,n_y}\, E_x^{-1} + t_{n_x,n_y}\, E_y + b_{n_x,n_y}\, E_y^{-1}\,)$$

is the local discretized differential operator at node (n_x, n_y). The Jacobi relaxation at a local node can be written as

$$u_{n_x,n_y}^{m+1} = J_{n_x,n_y}\, u_{n_x,n_y}^{m} + s_{n_x,n_y} \qquad n \geqslant 0 ,$$

where

$$J_{n_x,n_y} \equiv d_{n_x,n_y}^{-1} \, (r_{n_x,n_y} E_x + l_{n_x,n_y} E_x^{-1} + t_{n_x,n_y} E_y + b_{n_x,n_y} E_y^{-1})$$

is the local Jacobi relaxation operator. From the error point of view, we get

$$e_{n_x,n_y}^{m+1} = J_{n_x,n_y}\, e_{n_x,n_y}^{m} \qquad n \geqslant 0 .$$

If the input error function $e_{n_x,n_y}^{(m)}$ is the complex sinusoid $e^{i(\kappa_x x + \kappa_y y)}$, we have

$$J_{n_x,n_y}\, e^{i(\kappa_x x + \kappa_y y)} = \mu_{n_x,n_y}(\kappa_x ,\kappa_y)\, e^{i(\kappa_x x + \kappa_y y)}$$

where

$$\mu_{n_x,n_y}(\kappa_x ,\kappa_y) = d_{n_x,n_y}^{-1} \, (r_{n_x,n_y} e^{i\kappa_x h} + l_{n_x,n_y} e^{-i\kappa_x h} + t_{n_x,n_y} e^{i\kappa_y h} + b_{n_x,n_y} e^{-i\kappa_y h}).$$

Therefore, we may view $e^{i(\kappa_x x + \kappa_y y)}$ as an eigenfunction of J_{n_x,n_y} with eigenvalue $\mu_{n_x,n_y}(\kappa_x ,\kappa_y)$. The magnitude of $\mu_{n_x,n_y}(\kappa_x ,\kappa_y)$ provides some information on how the errors of different frequency components are smoothed out by the Jacobi relaxation process. This quantity can be computed as

$$| \mu_{n_x,n_y}(\kappa_x ,\kappa_y) | = \tag{8}$$

$$\frac{\left\{ [(r_{n_x,n_y} + l_{n_x,n_y})\cos\kappa_x h + (t_{n_x,n_y} + b_{n_x,n_y})\cos\kappa_y h]^2 + [(r_{n_x,n_y} - l_{n_x,n_y})\sin\kappa_x h + (t_{n_x,n_y} - b_{n_x,n_y})\sin\kappa_y h]^2 \right\}^{\frac{1}{2}}}{d_{n_x,n_y}}$$

Assuming that the coefficient functions are smooth so that

$$r_{n_x,n_y} - l_{n_x,n_y} = l_{n_x+1,n_y} - l_{n_x,n_y} = O(h) \text{ and } t_{n_x,n_y} - b_{n_x,n_y} = b_{n_x,n_y+1} - b_{n_x,n_y} = O(h),$$

then the two cosine terms in $| \mu_{n_x,n_y} |$ are the dominant terms.

The eigenvalue function $\mu_{n_x,n_y}(\kappa_x ,\kappa_y)$ is usually called the *frequency response* in signal processing [11] and the Jacobi relaxation operator can be viewed as a filtering process in the frequency domain. The frequency response function with the magnitude shown in (8), in fact, represents a 2-D notch filter instead of a lowpass filter. However, if the discretization space h is small enough and the waveforms are band-limited, this is not a significant problem. The reason is best explained from the Taylor's series approximation of a function $f(x)$, i. e.,

$$f(x) = f(x_0) + (x - x_0) f'(x_0) + \frac{(x - x_0)^2}{2!} f''(x_0) + \cdots$$

Supposing $f(x) = e^{i\kappa x}$ and $x = x_0 + h$, the high order terms are negligible only if the product kh is reasonably small, say, less than 1. That means that as long as the magnitude of wavevector k is bounded, we can always find a discretization spacing h which is fine enough so that the dimensionless frequencies $\theta_x = \kappa_x h$ and $\theta_y = \kappa_y h$ are always inside the unit circle in the (θ_x, θ_y) plane. In this region, the notch filter behaves like a lowpass filter. The lowpass filtering property makes the error at higher frequencies converge to zero faster than that at lower frequencies.

The eigenvalue with the largest magnitude is the dominant factor in the *asymptotic* convergence rate analysis, so that we will focus our attention on this quantity. Following the above discussion, we define the spectral radius ρ_{n_x,n_y} of J_{n_x,n_y} as the largest magnitude of $\mu_{n_x,n_y}(\kappa_x,\kappa_y)$, i.e.,

$$\rho_{n_x,n_y} \equiv \max_{\kappa_x,\kappa_y} | \mu_{n_x,n_y}(\kappa_x,\kappa_y) | .$$

For the symmetric positive definite matrix case, the magnitude of $\mu_{n_x,n_y}(\kappa_x,\kappa_y)$ is the largest at the lowest frequency $(\hat{\kappa}_x,\hat{\kappa}_y)$, since such a choice makes the dominant cosine terms of (8) as large as possible. Therefore, we obtain

$$\rho_{n_x,n_y} = | \mu_{n_x,n_y}(\hat{\kappa}_x,\hat{\kappa}_y) | .$$

The above procedure, known as local Fourier analysis of J_{n_x,n_y}, has two implicit assumptions. First, J_{n_x,n_y} is space-invariant. Secondly, the problem domain should be either extended to infinity or be rectangular but with periodic boundary conditions. In general, these two assumptions do not hold. As a consequence, ρ_{n_x,n_y} is a spatially varying function and is not equal to the spectral radius ρ of the original Jacobi relaxation matrix $J = D^{-1}(E+F)$. An important question to be answered is whether the knowledge of ρ_{n_x,n_y} can provide us with some information about ρ. Two observations may be of help. First, the *same* lowest frequency gives the spectral radii of all local Jacobi relaxation operators, so this frequency should play a role in determining the eigenfunction giving the spectral radius of the Jacobi relaxation matrix J. Furthermore, for a given low frequency $(\hat{\kappa}_x, \hat{\kappa}_y)$, ρ_{n_x,n_y} is a very smooth function in space. It is neither sensitive to variations of the coefficient functions nor sensitive to changes in the boundary conditions. For example, the values of ρ_{n_x,n_y} given by equations (3.2.7) and (8), computed for Dirichlet and periodic boundary conditions separately, are only slightly different under the assumption that the coefficient functions are smooth. Let $\bar{\rho} = \max\limits_{n_x,n_y} \rho_{n_x,n_y}$ and $\underline{\rho} = \min\limits_{n_x,n_y} \rho_{n_x,n_y}$. Then, it is our conjecture that ρ should be a quantity somewhere between $\underline{\rho}$ and $\bar{\rho}$. Usually, the difference between $\bar{\rho}$ and $\underline{\rho}$ is so small that *any* ρ_{n_x,n_y} can give us an estimate of ρ.

Notice that in order to determine the spectral radius of a local relaxation operator, we only have to know the lowest admissible Fourier component corresponding to the given boundary conditions, discussed in Section 3.4.1, and then to compute ρ_{n_x,n_y} according to (8).

3.4.3 Applying Fourier Analysis to the Local Relaxation Method

Let us reconsider the local relaxation method, i.e., equation (3.2.5). We divide the problem domain into red and black points and update one color at each time step. Suppose we start with the relaxation of the red points and, then update the black points. The local equations for the error can be written as

$$e_{n_x,n_y}^{m+1} = (1-\omega_{n_x,n_y})\, e_{n_x,n_y}^{m} + \omega_{n_x,n_y} J_{n_x,n_y} e_{n_x,n_y}^{m} \qquad n_x+n_y \ \text{even} \tag{9}$$

$$e_{n_x,n_y}^{m+1} = (1-\omega_{n_x,n_y})\, e_{n_x,n_y}^{m} + \omega_{n_x,n_y} J_{n_x,n_y} e_{n_x,n_y}^{m+1} \qquad n_x+n_y \ \text{odd} \tag{10}$$

Consider the neighborhood Ω_{n_x,n_y} of a point (n_x,n_y), where all J_{n_x,n_y}'s are approximately the same. Then, within Ω_{n_x,n_y}, we can combine (9) with (10) and rewrite equation (10) as

$$e_{n_x,n_y}^{m+1} = (1-\omega_{n_x,n_y})\, e_{n_x,n_y}^{m} + \omega_{n_x,n_y}(1-\omega_{n_x,n_y}) J_{n_x,n_y} e_{n_x,n_y}^{m} \tag{11}$$
$$+ \omega_{n_x,n_y}^2 J_{n_x,n_y}^2 e_{n_x,n_y}^{m} \qquad n_x+n_y \ \text{odd}.$$

Equations (9) and (11) describe, in fact, the evolution of two waves - the *red* and *black* waves in the local region Ω_{n_x,n_y}. To see this, let us first perform the local Fourier expansion of the values at the red and black points

$$e_{n'_x,n'_y} = \sum_{\kappa_x,\kappa_y} \hat{r}_{n_x,n_y}(\kappa_x,\kappa_y) e^{i(\kappa_x n'_x h + \kappa_y n'_y h)}, \qquad n'_x+n'_y \ \text{even}, \tag{12a}$$

$$e_{n'_x,n'_y} = \sum_{\kappa_x,\kappa_y} \hat{b}_{n_x,n_y}(\kappa_x,\kappa_y) e^{i(\kappa_x n'_x h + \kappa_y n'_y h)}, \qquad n'_x+n'_y \ \text{odd}, \tag{12b}$$

where $(n'_x,n'_y) \in \Omega_{n_x,n_y}$ and the summation includes all admissible frequencies for the problem domain Ω. Then, by substituting (12) into (9) and (11), we obtain the following relation between two successive iterations in the wavenumber domain,

$$\begin{bmatrix} \hat{f}^{m+1}_{n_x,n_y} \\ \hat{b}^{m+1}_{n_x,n_y} \end{bmatrix} = G_{n_x,n_y}(\omega_{n_x,n_y},\mu_{n_x,n_y}) \begin{bmatrix} \hat{f}^{m}_{n_x,n_y} \\ \hat{b}^{m}_{n_x,n_y} \end{bmatrix} ,$$

where

$$G_{n_x,n_y}(\omega_{n_x,n_y},\mu_{n_x,n_y}) = \begin{bmatrix} 1-\omega_{n_x,n_y} & \omega_{n_x,n_y}\,\mu_{n_x,n_y} \\ \omega_{n_x,n_y}\,(1-\omega_{n_x,n_y})\,\mu_{n_x,n_y} & 1-\omega_{n_x,n_y}+\omega^2_{n_x,n_y}\,\mu^2_{n_x,n_y} \end{bmatrix} . \tag{13}$$

and where μ_{n_x,n_y} is the eigenvalue for the eigenfunction $e^{i(\kappa_x x + \kappa_y y)}$ associated to the operator J_{n_x,n_y}. We use λ_{n_x,n_y} to denote the eigenvalues of $G_{n_x,n_y}(\omega_{n_x,n_y},\mu_{n_x,n_y})$.

In the space domain, we can associate (13) with a 2×2 matrix operator

$$G_{n_x,n_y}(\omega_{n_x,n_y},J_{n_x,n_y}) = \begin{bmatrix} 1-\omega_{n_x,n_y} & \omega_{n_x,n_y}\,J_{n_x,n_y} \\ \omega_{n_x,n_y}\,(1-\omega_{n_x,n_y})\,J_{n_x,n_y} & 1-\omega_{n_x,n_y}+\omega^2_{n_x,n_y}\,J^2_{n_x,n_y} \end{bmatrix} ,$$

which is called the *local relaxation* operator with relaxation factor ω_{n_x,n_y} at node (n_x,n_y). Its spectral radius $\rho[G_{n_x,n_y}(\omega_{n_x,n_y},J_{n_x,n_y})]$ is defined to be the spectral radius of the matrix $G_{n_x,n_y}(\omega_{n_x,n_y},\mu_{n_x,n_y})$ given by (13).

We know from (13) that $\mu_{n_x,n_y}(\kappa_x, \kappa_y)$ and $\lambda_{n_x,n_y}(\kappa_x, \kappa_y)$ are related via

$$\left| G_{n_x,n_y}(\omega_{n_x,n_y}, \mu_{n_x,n_y}) - \lambda_{n_x,n_y} I \right| = 0 ,$$

where | . | stands for the determinant. This gives

$$\lambda^2_{n_x,n_y} - (2 - 2\,\omega_{n_x,n_y} + \omega^2_{n_x,n_y}\,\mu^2_{n_x,n_y})\,\lambda_{n_x,n_y} + (1-\omega_{n_x,n_y})^2 = 0 .$$

Therefore, we have

$$\lambda_{n_x,n_y} = 1 - \omega_{n_x,n_y} + \frac{\omega^2_{n_x,n_y}\,\mu^2_{n_x,n_y}}{2} \pm \frac{\sqrt{\Delta}}{2} , \tag{14a}$$

where

$$\Delta = 4\,(1-\omega_{n_x,n_y})\,\omega^2_{n_x,n_y}\,\mu^2_{n_x,n_y} + \omega^4_{n_x,n_y}\,\mu^4_{n_x,n_y} . \tag{14b}$$

Let us consider the special case, $\omega_{n_x,n_y} = 1$, which corresponds to the Gauss-Seidel relaxation method. The eigenvectors of the 2×2 matrix $G_{n_x,n_y}(\omega_{n_x,n_y}, \mu_{n_x,n_y})$ are $(1,0)^T$ and $(1, \mu_{n_x,n_y})^T$ and the corresponding eigenvalues are 0 and $\mu^2_{n_x,n_y}$. This means that if we start with two sinusoidal waveforms at the same frequency but with different amplitudes, one of them, the red wave represented by the vector $(1,0)$, disappears in one step. The other wave remains and alternates between the red and black points thereafter. The ratio between the updated wave and the old wave is equal to the constant μ_{n_x,n_y}, so that the amplitude is reduced by a factor of $\mu^2_{n_x,n_y}$ per cycle.

The purpose of introducing the relaxation parameter ω_{n_x,n_y} is to make the eigenvalue $\lambda_{n_x,n_y}(\kappa_x,\kappa_y,\omega_{n_x,n_y})$ of the new operator G_{n_x,n_y} smaller than the eigenvalue $\mu_{n_x,n_y}(\kappa_x,\kappa_y)$ of the old operator J_{n_x,n_y}. For a fixed real $\mu_{n_x,n_y}(\kappa_x,\kappa_y)$, the relationship between λ_{n_x,n_y} and ω_{n_x,n_y} can be described by the root locus technique depicted in Figure 3.1.

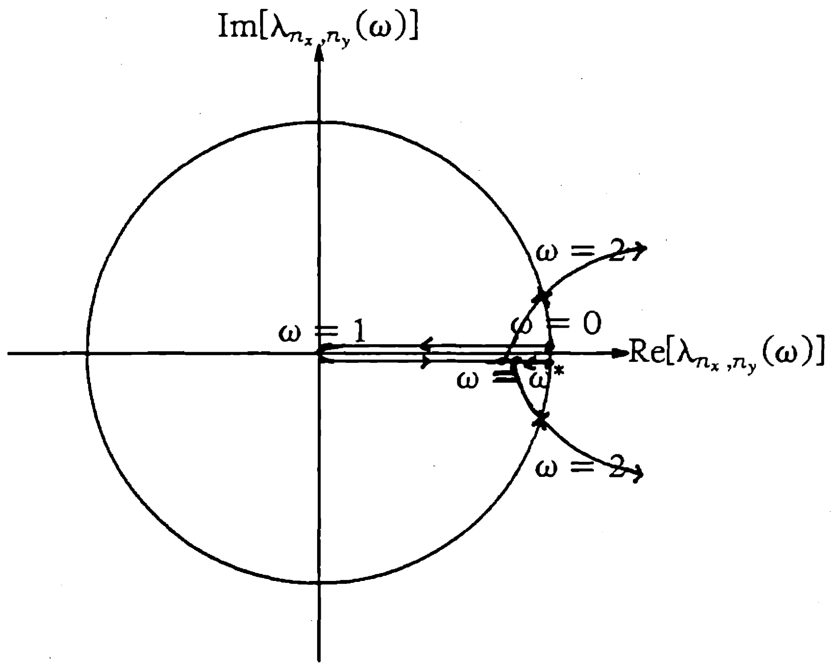

$\mathrm{Im}[\lambda_{n_x,n_y}(\omega)]$
$\mathrm{Re}[\lambda_{n_x,n_y}(\omega)]$
$\omega = 2$
$\omega = 1$
$\omega = 0$
$\omega = \omega^*$
$\omega = 2$

Figure 3.1: Root loci of $\lambda_{n_x,n_y}(\omega)$ with fixed μ_{n_x,n_y}

When $0 < \omega_{n_x,n_y} < 2$, the magnitude of λ_{n_x,n_y} is less than one. By Theorem 3.1, we know that if $0 < \omega_{n_x,n_y} < 2$ for all n_x, n_y then $\rho[G_W] < 1$ and the local relaxation algorithm converges. When $\Delta = 0$, the two eigenvalues $\lambda_{n_x,n_y,1}$ and $\lambda_{n_x,n_y,2}$ coincide, and the largest possible magnitude of these two eigenvalues,

$$\lambda_{n_x,n_y,m} \equiv \max(\ |\lambda_{n_x,n_y,1}|,\ |\lambda_{n_x,n_y,2}|\ ),$$

is minimized. The value of ω_{n_x,n_y} which sets $\Delta = 0$ is called the optimal relaxation factor with respect to a specific μ_{n_x,n_y} and is denoted by $\omega_{n_x,n_y,opt}(\mu_{n_x,n_y})$. By solving

$$\Delta = 4(1 - \omega_{n_x,n_y})\,\omega^2_{n_x,n_y}\,\mu^2_{n_x,n_y} + \omega^4_{n_x,n_y}\,\mu^4_{n_x,n_y} = 0$$

and requiring

$$0 < \omega_{n_x,n_y} < 2\ ,$$

we find that

$$\omega_{n_x,n_y,opt}(\mu_{n_x,n_y}) = \frac{2}{1 + (1 - \mu^2_{n_x,n_y})^{1/2}}\ . \tag{15}$$

The general relation between $\lambda_{n_x,n_y,m}$ and ω_{n_x,n_y} can be derived in a straightforward way from equation (14) and is given by

$$\lambda_{n_x,n_y,m} = \omega_{n_x,n_y} - 1 \qquad \omega_{n_x,n_y,opt}(\mu_{n_x,n_y}) \leqslant \omega_{n_x,n_y} < 2\ , \tag{16a}$$

$$\lambda_{n_x,n_y,m} = \left| \frac{\omega_{n_x,n_y}|\mu_{n_x,n_y}| + [\omega^2_{n_x,n_y}\mu^2_{n_x,n_y} + 4(1-\omega_{n_x,n_y})]^{1/2}}{2} \right|^2 \quad 0 < \omega_{n_x,n_y} \leqslant \omega_{n_x,n_y,opt}(\mu_{n_x,n_y}) \tag{16b}$$

The minimum value of all possible $\lambda_{n_x,n_y,m}$'s is, therefore, $\omega_{n_x,n_y,opt}(\mu_{n_x,n_y}) - 1$.

Since μ_{n_x,n_y} is a function of frequency, equation (15) implies that different frequencies require different optimal relaxation factors. However, we are allowed to choose only one ω_{n_x,n_y}, so we have to consider the overall performance, i.e., ω_{n_x,n_y} has to be selected so that the maximum of $\rho[G_{n_x,n_y}(\omega_{n_x,n_y}, J_{n_x,n_y})]$ over all frequencies is minimized.

Let ρ_{n_x,n_y} be the spectral radius of the local operator J_{n_x,n_y} and μ_{n_x,n_y} be an arbitrary eigenvalue of J_{n_x,n_y}. By definition, $|\mu_{n_x,n_y}| \leqslant \rho_{n_x,n_y}$, so we know that $\omega_{n_x,n_y,opt}(\mu_{n_x,n_y}) \leqslant \omega_{n_x,n_y,opt}(\rho_{n_x,n_y})$ from (15). Using the relation in (16), we reason as follows. If we choose $\omega_{n_x,n_y} = \omega_{n_x,n_y,opt}(\mu_{n_x,n_y})$, $\lambda_{n_x,n_y,m}(\mu_{n_x,n_y})$ achieves its minimum value of $\omega_{n_x,n_y,opt}(\mu_{n_x,n_y}) - 1$ but $\lambda_{n_x,n_y,m}(\rho_{n_x,n_y})$ is greater than $\omega_{n_x,n_y,opt}(\rho_{n_x,n_y}) - 1$. On the other hand, if $\omega_{n_x,n_y,opt}(\rho_{n_x,n_y})$ is chosen as the relaxation factor, both $\lambda_{n_x,n_y,m}(\mu_{n_x,n_y})$ and $\lambda_{n_x,n_y,m}(\rho_{n_x,n_y})$ are equal to $\omega_{n_x,n_y,opt}(\rho_{n_x,n_y}) - 1$. Comparing these two cases, the latter choice is the best scheme to minimize the spectral radius of $G_{n_x,n_y}(\omega_{n_x,n_y}, J_{n_x,n_y})$. This optimal value of ω_{n_x,n_y} is denoted as $\omega^*_{n_x,n_y}$, and is given by

$$\omega^*_{n_x,n_y} = \omega_{n_x,n_y,opt}(\rho_{n_x,n_y}) = \frac{2}{1 + (1 - \rho^2_{n_x,n_y})^{1/2}} .$$

This is exactly the same formula as suggested by Erhlich. The reason that this is a good choice is due to the fact that the eigenfunction with the largest eigenvalue of the Jacobi relaxation operator is the one corresponding to the lowest frequency component, and to the observation that the space varying relaxation parameter $\omega^*_{n_x,n_y}$ optimizes the convergence of this lowest frequency mode.

3.5 Local Relaxation for a 9-point Stencil Discretization

The above derivation applies to a 5-point stencil, which appears when we discretize a linear second-order elliptic PDE without the crossover term $\frac{\partial^2}{\partial x \, \partial y}$. If there is a crossover term, a finite difference discretization gives a 9-point stencil. In this section, we will propose a local relaxation scheme for a 9-point stencil, give a sufficient condition for its convergence, and use a Fourier analysis approach to explain the rule for selecting good local relaxation parameters. The approach is similar to that used in Section 3.4.

In order to make the presentation clear, we use a simple problem mentioned in [1] as an illustrative example. Consider the linear partial differential operator defined on $\Omega = [0,1] \times [0,1]$,

$$L = \frac{\partial^2}{\partial x^2} + a(x,y)\frac{\partial^2}{\partial x \, \partial y} + \frac{\partial^2}{\partial y^2}, \tag{1}$$

where $| a(x,y) | < 2$, with appropriate boundary conditions. The condition $| a(x,y) | < 2$ is required to guarantee that L is an elliptic operator. We also assume that $a(x,y)$ is sufficiently smooth so that it can be viewed as being approximately constant locally. The following discretization scheme is used

$$\frac{\partial^2}{\partial x^2} \longleftrightarrow \frac{E_x - 2 + E_x^{-1}}{h^2}, \quad \frac{\partial^2}{\partial y^2} \longleftrightarrow \frac{E_y - 2 + E_y^{-1}}{h^2},$$

$$\frac{\partial^2}{\partial x \, \partial y} \longleftrightarrow \frac{E_x E_y + E_x^{-1} E_y^{-1} - E_x^{-1} E_y - E_x E_y^{-1}}{4h^2}.$$

The local Jacobi relaxation operator J_{n_x,n_y} can be decomposed into two parts $J_{n_x,n_y,1}$ and $J_{n_x,n_y,2}$, i. e.

$$J_{n_x,n_y} = J_{n_x,n_y,1} + J_{n_x,n_y,2} ,$$

where

$$J_{n_x,n_y,1} = \frac{1}{4} (E_x + E_x^{-1} + E_y + E_y^{-1}) ,$$

$$J_{n_x,n_y,2} = \frac{1}{16} (a_{n_x+½,n_y+½} E_x E_y + a_{n_x-½,n_y-½} E_x^{-1} E_y^{-1} - a_{n_x-½,n_y+½} E_x^{-1} E_y - a_{n_x+½,n_y-½} E_x E_y^{-1}) .$$

and where

$$a_{n_x,n_y} = a(n_x h, n_y h) .$$

Suppose we use the red/black partitioning, then the $J_{n_x,n_y,1}$ operator couples nodes of different colors while the $J_{n_x,n_y,2}$ operator couples nodes of the same color. A four-color scheme which leads to a four-color SOR method has been proposed for this problem [1]. Here, we propose a different local relaxation scheme which uses only red/black partitioning, so that the iteration equations for the error in the local region can be written as

$$e_R^{m+1} = (1 - \omega_{n_x,n_y}) e_R^m + \omega_{n_x,n_y} \left[J_{n_x,n_y,1} e_B^m + J_{n_x,n_y,2} e_R^m \right] , \tag{2a}$$

$$e_B^{m+1} = (1 - \omega_{n_x,n_y}) e_B^m + \omega_{n_x,n_y} \left[J_{n_x,n_y,1} e_R^{m+1} + J_{n_x,n_y,2} e_B^m \right] , \tag{2b}$$

or, equivalently,

$$\begin{pmatrix} e_R^{m+1} \\ e_B^{m+1} \end{pmatrix} = G_{n_x,n_y}(\omega_{n_x,n_y}, J_{n_x,n_y,1}, J_{n_x,n_y,2}) \begin{pmatrix} e_R^m \\ e_B^m \end{pmatrix} ,$$

where

$$G_{n_x,n_y} (\omega_{n_x,n_y} , J_{n_x,n_y,1} , J_{n_x,n_y,2}) = \tag{3}$$

$$\begin{bmatrix} 1-\omega_{n_x,n_y} + \omega_{n_x,n_y} J_{n_x,n_y,2} & \omega_{n_x,n_y} J_{n_x,n_y,1} \\ \omega_{n_x,n_y}(1-\omega_{n_x,n_y}) J_{n_x,n_y,1} + \omega^2_{n_x,n_y} J_{n_x,n_y,1} J_{n_x,n_y,2} & 1-\omega_{n_x,n_y} + \omega_{n_x,n_y} J_{n_x,n_y,2} + \omega^2_{n_x,n_y} J^2_{n_x,n_y,1} \end{bmatrix} .$$

In general, the linear system of equations, $A\ u = s$ for a 9-point stencil which is obtained by discretizing an elliptic PDE with a crossover term can be decomposed as

$$A\,u = (D - E - F - C\,)u = D\,(I - L - U - V\,)u = s\ , \tag{4}$$

where A is an $N \times N$ real symmetric positive definite matrix and D, E, F, and C are diagonal, lower and upper triangular, and block diagonal matrices respectively. In (4), we view the 9-point stencil as the superposition of a standard 5-point stencil and of a 4-point stencil formed by the nodes at the four corners. The standard 5-point stencil is accounted for by $D - E - F$, while the remaining 4-point stencil due to the cross-over term is represented by the matrix $C = D\ V$. It is not hard to see that

$$E^T = F\ , \quad \text{and} \quad C^T = C\ .$$

According to the local relaxation method specified by (2), the matrix iterative equation in the error space becomes

$$e^{m+1} = (I - W\,L\,)^{-1}[\,(I - W\,) + W\,U + W\,V\,]\,e^m\ , \tag{5}$$

where W is a diagonal matrix formed by local relaxation parameters. The iteration matrix is therefore given by

$$G_W = (I - W\,L\,)^{-1}[\,(I - W\,) + W\,U + W\,V\,]\ .$$

A simple sufficient condition for the convergence of (5), or (2), can be obtained by generalizing Theorem 3.1. Following the same steps as in the proof of Theorem 3.1, we find that

$$\lambda = \frac{1 - \omega\,(\,1 - \alpha\,) + \omega\,\bar{z}}{1 - \omega\,z}\ ,$$

where λ, p is an arbitrary eigenvalue/eigenvector pair of G_W, $z = \dfrac{p^H\,D\,L\,p}{p^H\,D\,p}$, $\dfrac{1}{\omega} = \dfrac{p^H\,D\,W^{-1}\,p}{p^H\,D\,p}$, and $\alpha = \dfrac{p^H\,C\,p}{p^H\,D\,p}$. Since C is symmetric, α is a real number. Let $z = r\,e^{i\,\theta}$, then

$$| \lambda |^2 = 1 - \frac{\omega [2 - \omega (1 - \alpha)] (1 - \alpha - 2 r \cos \theta)}{(1 - \omega r \cos \theta)^2 + \omega^2 r^2 \sin^2 \theta} , \tag{6}$$

which is similar to (3.5). Now, consider

$$\begin{aligned}\alpha + 2 r \cos \theta &= \alpha + 2 \operatorname{Re} (z) = \alpha + \bar{z} + z \\ &= \frac{p^H D V p}{p^H D p} + \frac{p^H D L p}{p^H D p} + \frac{p^H D U p}{p^H D p} \\ &= 1 - \frac{p^H A p}{p^H D p} < 1 ,\end{aligned}$$

where the inequality is due to the fact that A and D are both positive definite. Furthermore, let us assume that $\alpha < 1$. In order to guarantee that $| \lambda | < 1$ for all possible eigenvalues, the sufficient condition becomes

$$0 < \omega_{\min} \leqslant \omega_{n_x, n_y} \leqslant \omega_{\max} < \frac{2}{1 - \alpha_{\min}} ,$$

where

$$\alpha_{\min} = \min_p \frac{p^H C p}{p^H D p} , \tag{7}$$

and where the minimization is over all eigenvectors p of the matrix G_W. Therefore, we have the following theorem.

Theorem 3.2 (Sufficient Condition for the Convergence of a Local Relaxation Procedure for a 9-point Stencil Discretization)
Suppose that A is an $N \times N$ real symmetric positive definite matrix. For the local relaxation procedure given by (4) and the constant $\alpha_{\min}$ defined by (7), if $0 < \omega_n < \frac{2}{1 - \alpha_{\min}}$ for $1 \leqslant n \leqslant N$, then $\rho[G_W] < 1$ and the iterative algorithm converges.

Note that if there is no crossover term, the matrix C is zero and α is also zero. In this case, the 9-point stencil reduces to a 5-point stencil and Theorem 3.2 reduces to Theorem 3.1. Therefore, our proposed local relaxation scheme for a 9-point stencil, equation (5), is a natural generalization of the conventional SOR method for a 5-point stencil, specified by equation (3.3.2).

In the above derivation, we have used the assumption that $\alpha = \frac{p^H C p}{p^H D p}$ is less than 1 for any eigenvector p of the matrix G_W. Now, let us estimate the value of α by examining the example given by (1). For simplicity, we consider the special case where $a(x,y) = a$ is constant and assume that the boundary conditions are *periodic*, i.e. $u(0,y) = u(1,y)$ for $0 \leqslant y \leqslant 1$, $u(x,0) = u(x,1)$ for $0 \leqslant x \leqslant 1$. In this case, the eigenvectors p of G_W can be found in closed form and are given by one of the following two dimensional arrays,

$\sin(\kappa_x n_x h + \kappa_y n_y h)$, $\cos(\kappa_x n_x h + \kappa_y n_y h)$, $\sin(\kappa_x n_x h - \kappa_y n_y h)$ and $\cos(\kappa_x n_x h - \kappa_y n_y h)$, where n_x and n_y range from 1 to $\sqrt{N}$, $h = \frac{1}{\sqrt{N}}$, and κ_x, κ_y are multiples of 2π. Then, after some computations, we find that

$$\alpha(p) = \frac{p^H C p}{p^H D p} = \pm \frac{a}{4} \sin\kappa_x h \sin\kappa_y h \quad \text{and} \quad |\alpha(p)| < \frac{|a|}{4} \text{ for all } p .$$

Therefore, if we choose ω_n between 0 and $\frac{8}{4+|a|}$, the local relaxation algorithm for this particular problem will converge. However, this choice is too conservative to give a good convergence rate when $|a|$ is close to 2.

Generally speaking, two types of errors arise in the numerical solution of elliptic PDEs by iterative methods. The first of these is caused by the error between the initial guess and the true solution. The other is the numerical rounding error due to the finite precision arithmetic. The first error is usually concentrated in the low frequency region, whereas the second can exist at all possible frequencies. The numerical rounding error is usually so small that it can be ignored, provided it does not grow with the number of iterations. Thus, the error smoothing primarily aims at reducing errors in the low frequency region where the initial guess errors are substantial.

Let us temporarily ignore the numerical rounding error and focus on the initial guess error only. In order to guarantee the convergence of all components in the low frequency region, we need only to select

$$\alpha_{\min} := \alpha_{\min}^{L} \equiv -\frac{|a|}{4}\sin(\tilde{\kappa}_x h)\sin(\tilde{\kappa}_y h)$$

where $\tilde{\kappa}_x$ and $\tilde{\kappa}_y$ are the largest frequencies of interest. Usually the mesh is so fine that $\alpha_{\min}^{L}$ is of order $O(h^2)$. Although this conclusion is obtained from a simple example, it seems reasonable to believe that $\alpha_{\min}^{L}$ is also of order $O(h^2)$ for more general second-order elliptic PDEs with space-varying coefficients and other boundary conditions.

The remaining problem is to select a set of local relaxation parameters such that the iterative algorithm converges as quickly as possible in the low frequency region. We can use the Fourier analysis approach introduced in Section 3.4 to analyze the local 2×2 matrix operator $G_{n_x,n_y}(\omega_{n_x,n_y}, J_{n_x,n_y,1}, J_{n_x,n_y,2})$ given by (3).

Let $\mu_{n_x,n_y,1}(\kappa_x,\kappa_y)$, $\mu_{n_x,n_y,2}(\kappa_x,\kappa_y)$ and $\mu_{n_x,n_y}(\kappa_x,\kappa_y)$ be eigenvalues of $J_{n_x,n_y,1}$, $J_{n_x,n_y,2}$ and J_{n_x,n_y} respectively. Following the procedure used before, we find that the optimal local relaxation factor for (1) is

$$\omega_{n_x,n_y}^{*} = \frac{2}{1-\epsilon_{n_x,n_y} + [\,(1-\epsilon_{n_x,n_y})^2 - \rho_{n_x,n_y}^2\,]^{1/2}} \tag{8}$$

where $\epsilon_{n_x,n_y} = \mu_{n_x,n_y,2}(\hat{\kappa}_x,\hat{\kappa}_y)$, $\rho_{n_x,n_y} = \mu_{n_x,n_y,1}(\hat{\kappa}_x,\hat{\kappa}_y)$ and $(\hat{\kappa}_x,\hat{\kappa}_y)$ is the mode which maximizes $\mu_{n_x,n_y} = \mu_{n_x,n_y,1} + \mu_{n_x,n_y,2}$. The spectral radius of the local relaxation operator G_{n_x,n_y} is

$$\lambda_{n_x,n_y}\left(G_{n_x,n_y}\left(\omega^*_{n_x,n_y} , J_{n_x,n_y,1} , J_{n_x,n_y,2} \right)\right) = \omega^*_{n_x,n_y}\left(1 + \epsilon_{n_x,n_y} \right) - 1 . \qquad (9)$$

Typical values for $\omega^*_{n_x,n_y}$ and λ_{n_x,n_y} can be obtained by considering example (1) with Dirichlet boundary conditions, which is of more interest compared to the periodic boundary conditions in practice. Since $a(x,y)$ is smooth, we can approximate $J_{n_x,n_y,2}$ by

$$J_{n_x,n_y,2} \approx \frac{a_{n_x,n_y}}{16}\left(E_x E_y + E_x^{-1} E_y^{-1} - E_x^{-1} E_y - E_x E_y^{-1} \right) .$$

The values of $\mu_{n_x,n_y,1}(\kappa_x,\kappa_y)$, $\mu_{n_x,n_y,2}(\kappa_x,\kappa_y)$, and $\mu_{n_x,n_y}(\kappa_x,\kappa_y)$ are

$$\mu_{n_x,n_y,1}(\kappa_x,\kappa_y) = \frac{1}{2}\left(\cos\kappa_x h + \cos\kappa_y h \right)$$

$$\mu_{n_x,n_y,2}(\kappa_x,\kappa_y) = \pm \frac{a_{n_x,n_y}}{8}\left[\cos(\kappa_x+\kappa_y)h - \cos(\kappa_x-\kappa_y)h \right]$$

$$= + \frac{a_{n_x,n_y}}{4} \sin\kappa_x h \ \sin\kappa_y h$$

and

$$\mu_{n_x,n_y}(\kappa_x,\kappa_y) = \frac{1}{2}\left(\cos\kappa_x h + \cos\kappa_y h \right) \pm \frac{a_{n_x,n_y}}{4} \sin\kappa_x h \ \sin\kappa_y h \ .$$

In the low frequency region, $J_{n_x,n_y,1}$ is a lowpass filter, while $J_{n_x,n_y,2}$ is a highpass filter. J_{n_x,n_y} is similar to $J_{n_x,n_y,1}$ in this region, because $\mu_{n_x,n_y,2}$ is almost zero for very low frequencies. Therefore, we can view $J_{n_x,n_y,2}$ as a *perturbation*. Thus, J_{n_x,n_y} is a perturbed lowpass filter. Its spectral radius $\rho_{n_x,n_y}(J_{n_x,n_y})$ is determined by the lowest admissible frequency. For this particular problem, the lowest admissible frequency mode turns out to be $(\hat{\kappa}_x,\hat{\kappa}_y) = (\pi,\pi)$. Therefore,

$$\epsilon_{n_x,n_y} = \frac{|a_{n_x,n_y}|}{4} \sin^2 \pi h \ , \quad \rho_{n_x,n_y} = \cos \pi h \ .$$

Substituting these values back into (8) and (9), we find

$$\omega^*_{n_x,n_y} \approx \frac{2}{1 + (1+\frac{|a_{n_x,n_y}|}{2})^{1/2} \pi h} \approx 2 [1 - (1+\frac{|a_{n_x,n_y}|}{2})^{1/2} \pi h] , \tag{10a}$$

$$\lambda_{n_x,n_y} \approx 1 - (4+2|a_{n_x,n_y}|)^{1/2} \pi h \ . \tag{10b}$$

In general, $\alpha^L_{\min}$ is of $O\,(\,h^2\,)$ so that for sufficiently small values of h, the above optimal relaxation parameters satisfy the sufficient condition of Theorem 3.2. Hence, the convergence of the local relaxation method in the low frequency region is guaranteed and the convergence rate can be estimated by examining the spectral radius of the local relaxation operator given by (10b).

Now let us go back to the effect of numerical rounding errors. The optimal relaxation parameters $\omega^*_{n_x,n_y}$ given by (10a) may be outside the convergence range defined by 0 and $\frac{2}{1-\alpha(p)}$ for some eigenvectors p corresponding to high frequency error components. Therefore, we expect that the error in the high frequency region will grow. The error growth rate is problem dependent and can be analyzed by Fourier analysis. If the error growth rate is so slow that it does not effect the answer much, we can stick to a single set of optimal local relaxation parameters. On the other hand, if the error growth rate is relatively large, we may use two sets of local relaxation parameters. One set aims at reducing the low frequency error quickly and the other set, formed by smaller values of ω_{n_x,n_y}, is used to smooth the high frequency error once in a while so that the rounding errors do not accumulate. This mixed scheme should perform much better than a scheme using a single set of conservative local relaxation parameters. However, the optimal scheduling of these two sets of local relaxation parameters is still unknown. We believe that it depends on the problem to be solved. Some numerical experiments will be needed to gain a better understanding of this issue.

3.6 Convergence Rate Analysis and Numerical Examples

3.6.1 Convergence Rate Analysis for the Linear Constant-Coefficient Case

For a linear constant coefficient PDE defined on a unit square with Dirichlet boundary conditions, the spectral radii of all local Jacobi operators are the same, and thus all local relaxation parameters and the spectral radii of all local relaxation operators are the same, i.e. for all n_x, n_y

$$\rho_{n_x,n_y} = \rho\ , \quad \omega_{n_x,n_y} = \omega\ , \quad \lambda_{n_x,n_y} = \lambda\ . \tag{1}$$

In this case, the local relaxation method is the same as SOR.

The asymptotic convergence rate of an arbitrary global iterative operator P, denoted by $R_\infty(P)$, is defined as [17]

$$R_\infty(P) = -\ln \rho(P)\ .$$

Under the conditions (1), the asymptotic convergence rate is also given by

$$R_\infty(P) = -\ln \rho(P_{n_x,n_y})$$

where P_{n_x,n_y} is the local relaxation operator of P.

In particular, let us use the Poisson equation on the unit square with Dirichlet boundary conditions as an example. The local Jacobi operator for this particular problem is

$$J_{n_x,n_y} = \frac{1}{4}(E_x + E_x^{-1} + E_y + E_y^{-1})\ .$$

Applying Fourier analysis to J_{n_x,n_y}, we find that the spectral radius of J_{n_x,n_y} is $\cos \pi h$, where h is the grid spacing. The global asymptotic convergence rate of the Jacobi method, the Gauss-Seidel method, and the local relaxation method can be computed as [8]

$$R_\infty(\text{ Jacobi }) = -\ln \rho(J_{n_x,n_y}) = -\ln \cos\pi h \approx \frac{1}{2}\pi^2 h^2 .$$

$$R_\infty(\text{ Gauss–Seidel }) = -\ln \lambda[G_{n_x,n_y}(1,J_{n_x,n_y})] = -\ln \cos^2\pi h$$

$$\approx \pi^2 h^2 ,$$

$$R_\infty(\text{ local relaxation }) = -\ln \lambda[(G_{n_x,n_y}(\omega^*,J_{n_x,n_y})] = -\ln(\omega^* - 1)$$

$$= -\ln\left(\frac{2}{1+(1-\cos^2\pi h)^{1/2}} - 1\right) \approx 2\pi h .$$

Therefore, the number of iterations is proportional to $O(\frac{1}{h})$, i.e. $O(\sqrt{N})$, for the local relaxation method.

3.6.2 Numerical Examples

For general space-varying coefficient PDEs, it is difficult to analyze the convergence rate as shown in Section 3.6.1. So, a simple numerical example is used to illustrate the convergence rate of the local relaxation method for solving space-varying coefficient PDEs. The convergence rates of the SOR and CG methods are also shown for the purpose of comparison. For the SOR and CG methods, the Ellpack software package [13] was used.

The example chosen is

$$e^{xy}\frac{\partial^2 u}{\partial x^2} + e^{-xy}\frac{\partial^2 u}{\partial y^2} + e^{xy}\, y\,\frac{\partial u}{\partial x} - e^{-xy}\, x\,\frac{\partial u}{\partial y} + \frac{1}{1+x+y}u \tag{2}$$

$$= e^{2xy}\sin\pi y\,[(2y^2-\pi^2)\sin\pi x + 3\pi y\cos\pi x] + \pi\sin\pi x(x\cos\pi y - \pi\sin\pi y)$$

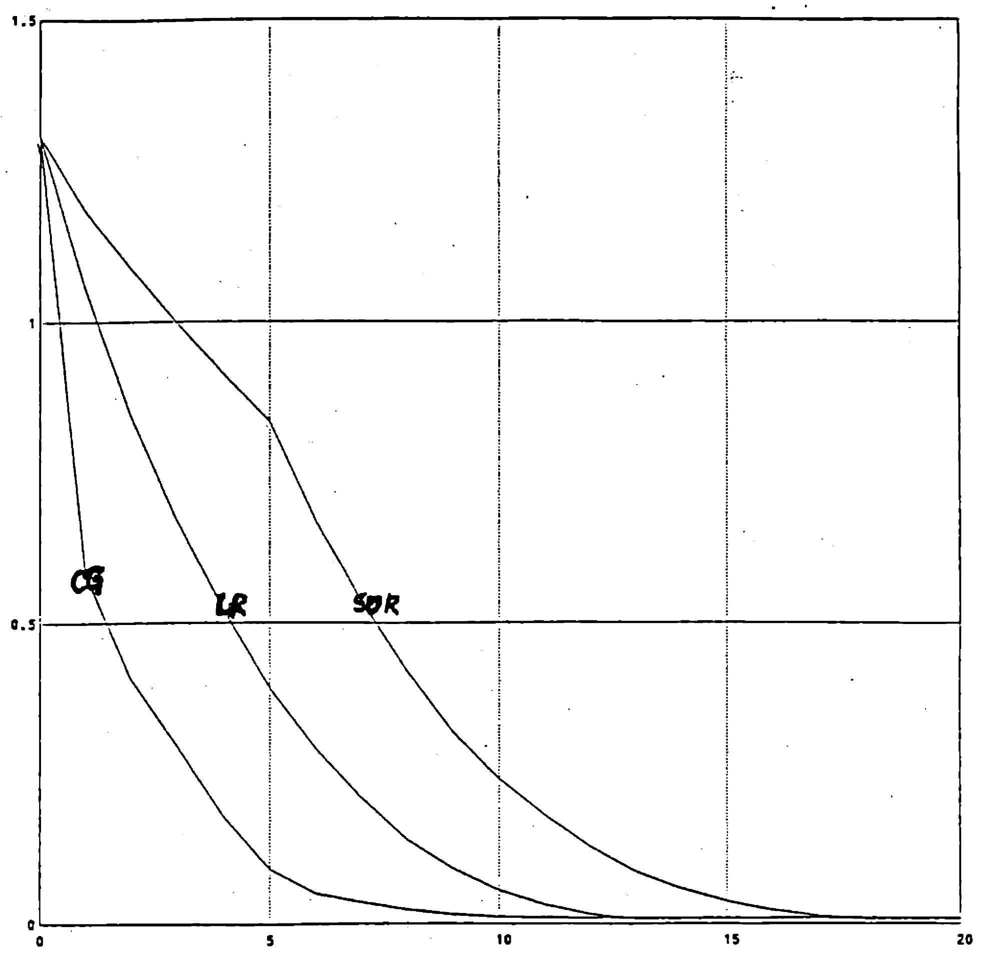
$$+ \frac{e^{xy}\sin\pi x\,\sin\pi y}{1+x+y}$$

on the unit square, with the boundary conditions

$$u(x,y) = 0, \quad \text{for } x=0, x=1, y=0, \text{and } y=1, \tag{3}$$

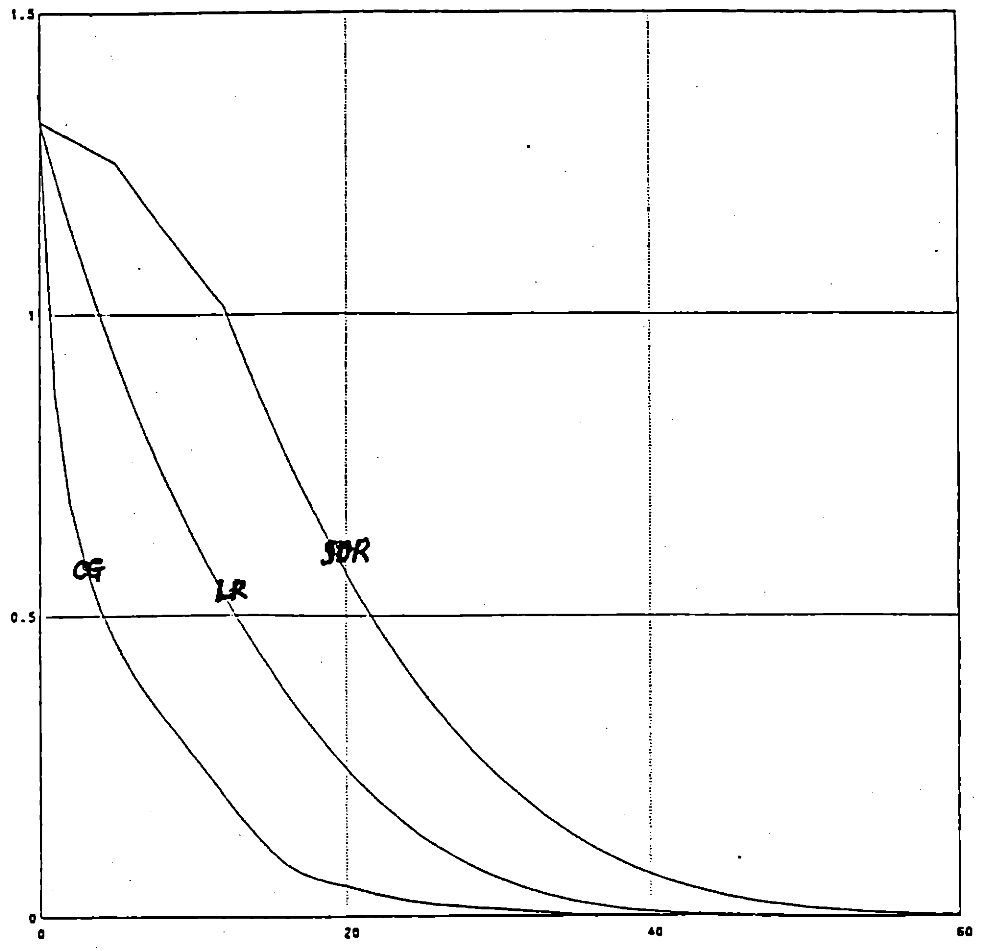
and its solution is $e^{xy}\sin\pi x\,\sin\pi y$. Although equation (2) does not have the same form as (2.1), it is easy to verify that the discretized matrix is still symmetric positive definite so that Theorem 3.1 applies here. The lowest frequency mode for this problem is $(\hat{\kappa}_x, \hat{\kappa}_y) = (\pi, \pi)$ because of the Dirichlet boundary conditions.

For this test problem, three 5-point discretization schemes are used with grid spacings $\frac{1}{10}$, $\frac{1}{30}$, and $\frac{1}{50}$. Starting from the initial guess $u^0(x,y) = 0$ for all grid points, the maximum errors at each iteration are plotted in Figure 3.2. The results indicate that on a single processor the convergence rate of the local relaxation method is better than that of the SOR method and worse than that of the CG method. However, on a mesh-connected array the local relaxation takes constant time for each iteration while the CG and SOR methods take $O(\sqrt{N})$ time per iteration, so that the local relaxation method is much faster (see Chapter 5). We also note that the number of iterations required for the local relaxation method is proportional to $\sqrt{N}$. This is consistent with the analysis of the previous section.

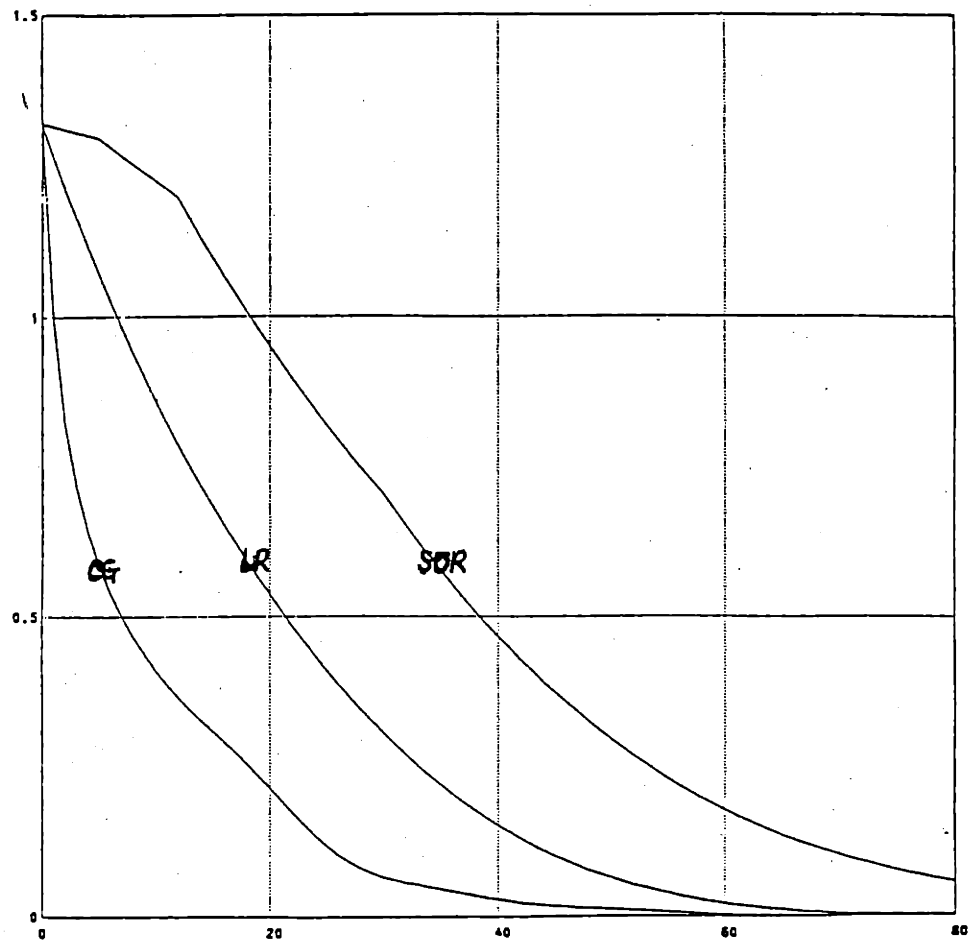


(a)

Figure 3.2: Computer simulation results for given example (2)-(3) with (a) 11×11, (b) 31×31, and (c) 51×51 grids.The x-axis is the number of iterations and the y-axis is the maximum error at each iteration.



(b)



(c)

3.7 Conclusions and Extensions

The local relaxation method includes two important steps. The first is to determine the admissible lowest frequencies using boundary condition information. The second is to approximate the PDE operator locally by a linear finite difference operator, divide the nodes into red and black points, and form a locally accelerated successive over-relaxation (local relaxation) operator. In previous discussions, some ideal assumptions were made so that the analysis and design of the local relaxation algorithm become very simple. However, we may encounter several difficulties in applying the local relaxation method directly to real world problems.

Under the assumption that the problem domain is a unit square and that the boundary condition operator is constant along each edge, the procedure for determining the lowest admissible frequencies is straightforward. These assumptions make the basis functions separable and easy to analyze. However, in practice, the above assumptions may not hold. The problem domain is usually of irregular shape and the boundary condition operators may have space-varying coefficients. As a consequence, it is considerably more difficult to find the lowest frequency error component than for the case we have considered in this chapter.

The second difficulty is related to the construction of the local relaxation operator. If the coefficients of a PDE operator have some discontinuities in some region, the Jacobi relaxation operator is not smooth over the region with discontinuous coefficients. In this case, the determination of the optimal local relaxation factors for such abruptly changing operators is still an open question.

3.8 References

[1] L. Adams and J. M. Ortega, "A Multi-Color SOR Method for Parallel Computation," ICASE report, 82-9, Apr. 1982.

[2] S. H. Bokhari, "On the Mapping Problem," *IEEE transaction on Computers*, vol. C-30, no. 3, pp. 207-214, Mar. 1981.

[3] E. F. Botta and A. E. P. Veldman, "On Local Relaxation Methods and Their Application to Convection-Diffusion Equations," *Journal of Computational Physics*, vol. 48, pp. 127-149, 1981.

[4] A. Brandt , "Multi-level Adaptive Solutions to Boundary-value Problems," *Mathmatics of Computation*, vol. 31, no. 138, pp. 333-390, Apr. 1977.

[5] L. W. Erhlich, "An Ad Hoc SOR Method," *Journal of Computational Physics*, vol. 44, pp. 31-45, 1981.

[6] L.W. Erhlich, "The Ad-hoc SOR Method: a Local Relaxation Scheme," in *Elliptic Problem Solvers II*, Academic Press, 1984, pp. 257-269.

[7] D. Gannon, "On Mapping Non-uniform PDE Structures and Algorithms onto Uniform Array Architectures," in *Proceeding of International Conference on Parallel Processing*, 1981.

[8] L. A. Hageman and D. M. Young, *Applied Iterative Methods*. New York, N.Y. : Academic Press, Inc. , 1981.

[9] C.-C. Kuo, "Parallel Algorithms and Architectures for Solving Elliptic Partial Differential Equations," MIT Technical Report (LIDS-TH-1432), Jan. 1985.

[10] C. A. Mead and L. A. Conway, *Introduction to VLSI Systems*. Reading, Mass : Addison-Wesley , 1980.

[11] A. V. Oppenheim and R. W. Schafer, *Digital Signal Processing*. Englewood Cliffs, N.J. : Prentice-Hall, Inc. , 1975.

[12] D. W. Peaceman, *Fundamentals of Numerical Reservoir Simulation*. New York, N.Y.: Elsevier North-Holland Inc., 1977.

[13] J. R. Rice, "ELLPACK : Progress and Plans," in *Elliptic Problem Slovers*, ed. M. H. Schultz, New York, N. Y.: Academic Press, Inc., 1981, pp. 135-162.

[14] P. J. Roache, *Computational Fluid Dynamics*. Albuquerque, New Mexico: Hermosa Publishers, 1976.

[15] H. L. Stone, "Iterative Solution of Implicit Approximations of Multidimensional Partial Differential Equations," *SIAM Journal on Numerical Analysis*, vol. 5, no. 3, pp. 530-558.

[16] K. Stuben and U. Trottenberg, "Multigrid Methods : Fundamental Algorithms, Model Problem Analysis, and Applications," in *Multigrid Methods*, ed. U. Trottenberg, New York, N.Y.: Springer-Verlag, 1982.

[17] R. S. Varga, *Matrix Iterative Analysis*. Englewood Cliffs, N.J. : Prentice-Hall, Inc. , 1962.

[18] D. M. Young , "Iterative Methods for Solving Partial Differential Equations of Elliptic Type ," Doctoral Thesis , Harvard University , 1950.

Chapter 4 : Two-level Four-color SOR Method

4.1 Introduction

The successive overrelaxation (SOR) method introduced in the early 1950's is an effective scheme for accelerating basic relaxation methods such as the Jacobi and Gauss-Seidel iterations [7][12]. The acceleration effect relies on the properties of a special class of matrices known as the p-cyclic [11] or the *consistently ordered* [13] matrices. By discretizing elliptic PDEs with finite difference schemes, we often obtain sparse matrix equations where the matrix is consistently ordered. Consequently, the SOR method has a wide range of applications.

Three types of approaches, the *pure space* domain, *semi-frequency* domain and *pure frequency* domain approaches, can be used to study the SOR iteration for solving elliptic PDEs. Young's work [12] [13] is a typical example of the pure space domain approach. This approach starts from an expression for the SOR iteration in the space domain. Then, under some conditions such as consistent ordering and property A [13], an argument based on matrix algebra is used to find a relation between the optimal relaxation parameter for the SOR method and the spectral radius of the Jacobi relaxation matrix. This procedure does not exploit any information provided by the eigenfunctions of the iteration matrix, so that the result obtained by this approach can be applied to a large class of problems such as space-varying coefficient PDEs on irregular domains.

The approach used in [2], [7], [8], [9] and [10] can be viewed as a semi-frequency domain approach, which adopts the space domain formulation but uses a frequency domain, or Fourier, analysis technique. This approach still starts from a fixed expression for the SOR iteration in the space domain. Then, under the assumption that the PDEs have constant coefficients and are defined on a rectangular domain with Dirichlet or periodic boundary conditions, sinusoidal functions turn out to be eigenfunctions of the discretized system of equations [7][10]. Hence, the system of equations can be decoupled by using these functions as a basis and, as a consequence, each frequency can be considered separately. This approach, although only rigorous for a restricted class of problems, provides a simple explanation of how the SOR method works.

A common feature of the above two approaches is that an SOR iteration form in the space domain has to be specified a priori. For simple cases such as for a 5-point discretization of the Poisson equation, most reasonable SOR iteration forms lead to an analysis in which the optimal relaxation parameter can be determined in closed form. However, for more complicated cases, such as the 9-point stencil case, it is not easy to specify in advance an iteration form whose analysis will be easy [1][3]. A class of 9-point stencil SOR iteration forms in the space domain was analyzed by Adams, LeVeque and Young [2]. Since the iteration matrices obtained from these forms are not consistently ordered, the traditional SOR theory cannot be applied for determining the optimal relaxation parameter. Hence, Adams et al. used a separation of variables technique to study the eigenvalues and eigenfunctions of the system of equations, and showed that the optimal relaxation parameter can be determined by solving a quartic equation [2].

In this chapter, we study the same problem, i.e., we develop an SOR method for the 9-point discretization of the Poisson equation. However, we use a pure frequency domain approach. This approach makes use of the traditional SOR theory for p-cyclic matrices in the frequency domain. We first divide grid points into 4 colors. By assuming that the PDE has constant coefficients and is defined on a square, we can apply Fourier analysis to each color so that a 4-color matrix equation can be obtained in the frequency domain. The 4-color matrix is block diagonal with 4×4 matrix blocks along the diagonal. Each of these blocks relates Fourier components of the 4 colors at a single frequency. If we partition the 4×4 matrix associated to a fixed frequency into four 2×2 blocks, the block partitioned matrix is 2-cyclic. Therefore, at a first level, we can use a standard block SOR iteration to accelerate the block Jacobi relaxation. Then, to decouple values of two different colors within the same block, we have to invert a 2×2 matrix. This can be easily accomplished by using a point SOR iteration at a second level. Once the appropriate 2-level SOR iteration form is determined in the frequency domain, it is straightforward to transform it back to the space domain. The details are described in Section 4.3.

This pure frequency domain approach yields a new 2-level 4-color SOR method which is completely different from the single-level SOR method studied in [2]. For a fixed color ordering, the 2-level 4-color SOR method can be described as follows. At a first level, the red and orange points, and then the black and green points are treated as groups, and a block SOR iteration is performed on these two groups. At a second level, another SOR iteration is used to decouple values at the red and orange points, and then at the black and green points. The optimal relaxation parameters ω_b^* and ω_p^* at the two iteration levels can be expressed in closed-form. The 2-level SOR method is easy to implement, and its spectral radius is of the form $1 - Ch$, where C is a constant comparable to the one obtained in [2].

The chapter is organized as follows. In Section 4.2, we use a simple 1-D 2-color SOR method to demonstrate our pure frequency domain approach. Section 4.3 describes the main result of this chapter, i.e., the 2-D 2-level 4-color SOR algorithm for a 9-point discretization of the Poisson equation. Then, in Section 4.4, we show that the conventional 2-D single-level 2-color SOR method for the 5-point stencil case is a degenerate case of the general 2-level 4-color scheme. Closed-form formulas for the optimal relaxation parameters ω_b^* and ω_p^* corresponding to the 2 iteration levels are obtained in Section 4.5, where the convergence rate of the 2-level SOR method is also analyzed. Finally, some numerical results are presented in Section 4.6.

4.2 1-D 2-color SOR Method

In this section, we consider a simple 1-D model problem and show how the 2-color SOR method can be derived from the Jacobi iteration method by first transforming the problem to the frequency domain and then introducing the relaxation parameter ω inside the frequency-domain iteration matrix. Although the final result is well known, the approach we are taking is new and provides some new insight. The same approach will be used to develop a 2-level iteration method in the next section.

4.2.1 Problem Formulation

Consider the discrete 1-D Poisson equation on [0,1] with grid spacing h

$$\frac{1}{h^2}(u_{n-1} - 2u_n + u_{n+1}) = f_n \ , \qquad n = 1, 2, \cdots, N-1 \ ,$$

where u_0, u_N are given, and $N = \frac{1}{h}$. Suppose we divide the problem domain into red and black points corresponding respectively to points with even and odd indices. With this partitioning, the Jacobi iteration method takes the form

$$u_n^{m+1} = \frac{1}{2}(u_{n-1}^m + u_{n+1}^m - 2h^2 f_n) \qquad n \ \text{ even} \ ,$$

$$u_n^{m+1} = \frac{1}{2}(u_{n-1}^m + u_{n+1}^m - 2h^2 f_n) \qquad n \ \text{ odd} \ .$$

Denote the exact solution by $\bar{u}_n$ and define the error as $e_n^m = u_n^m - \bar{u}_n$. Then, the error equations can be written as

$$\begin{aligned} e_n^{m+1} &= \frac{1}{2}(e_{n-1}^m + e_{n+1}^m) && n \ \text{ even} \ , \\ e_n^{m+1} &= \frac{1}{2}(e_{n-1}^m + e_{n+1}^m) && n \ \text{ odd} \ , \end{aligned} \tag{1}$$

with $e_0 = e_N = 0$.

Since (1) is a system of linear constant-coefficient equations with homogeneous boundary conditions, the eigenfunctions of this system are given by $\sin(k\pi nh)$, where $k = 1, 2, \cdots, (N-1)$. These functions form a basis, so that

$$\begin{aligned} r_n^m &= \sum_{k=1}^{N-1} \hat{r}_k^m \sin(k\pi nh) \quad 1 \leqslant n \leqslant N-1 \ , \\ b_n^m &= \sum_{k=1}^{N-1} \hat{b}_k^m \sin(k\pi nh) \quad 1 \leqslant n \leqslant N-1 \ , \end{aligned} \tag{2}$$

where the coefficients $\hat{r}_k^m$ and $\hat{b}_k^m$ are chosen such that

$$\begin{aligned} r_n^m &= e_n^m \qquad n \text{ even} \ , \\ b_n^m &= e_n^m \qquad n \text{ odd} \ . \end{aligned} \tag{3}$$

In other words, r_n^m and b_n^m are two sequences which coincide with the errors at red and black points respectively. They can be viewed as interpolations of the errors at the red and black points to all grid points. Note that there are $2(N-1)$ undetermined coefficients in (2) and only $N-1$ constraints in (3). Since (2) and (3) form a under-determined system of equation, there are many ways to choose $\hat{r}_k^m$ and $\hat{b}_k^m$. However, the actual values of these coefficients are not important. We are primarily concerned with how they evolve as the iteration proceeds.

Consider the error dynamics relating r_n and b_n,

$$\begin{aligned} r_n^{m+1} &= \frac{1}{2}\left(b_{n-1}^m + b_{n+1}^m \right) \qquad 1 \leqslant n \leqslant N-1 \ , \\ b_n^{m+1} &= \frac{1}{2}\left(r_{n-1}^m + r_{n+1}^m \right) \qquad 1 \leqslant n \leqslant N-1 \ . \end{aligned} \tag{4}$$

Although (4) contains more information than (1), all the information contained in (1) is preserved by (4) and the dynamic behavior of (1) can be obtained by studying the dynamic behavior of (4). Conceptually, (4) is easier to analyze than (1) since it is a spatially invariant system for both red and black colors.

By substituting (2) inside (4), for $k = 1, 2, ..., N-1$, we have

$$\begin{pmatrix} \hat{r}_k^{m+1} \\ \hat{b}_k^{m+1} \end{pmatrix} = B(k) \begin{pmatrix} \hat{r}_k^m \\ \hat{b}_k^m \end{pmatrix} , \tag{5}$$

where

$$B(k) = \begin{bmatrix} 0 & \cos(k\pi h) \\ \cos(k\pi h) & 0 \end{bmatrix} \tag{6}$$

is called the *Jacobi iteration matrix* for the frequency $k\pi$, which has two eigenvalues

$$\mu_k = \pm \cos(k\pi h) .$$

Intuitively speaking, we use the fact that the sinusoidal functions are eigenfunctions of the linear system (4) so that, by changing the coordinate from the space domain to the frequency domain, we are able to decompose the loosely coupled system (4) into a decoupled system which is a block diagonal matrix containing many 2×2 matrices along the diagonal.

Since the spectral radius of $B(k)$ is less than 1 for any k, the iteration (5) converges. Consequently, the asymptotic values $\hat{r}_k^\infty$ and $\hat{b}_k^\infty$ obtained by this iteration procedure are $\hat{r}_k^\infty = \hat{b}_k^\infty = 0$, and (5) can be viewed as obtained by solving the linear system

$$A(k) \begin{pmatrix} \hat{r}_k^\infty \\ \hat{b}_k^\infty \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} , \quad \text{with} \quad A(k) = \begin{bmatrix} 1 & -\cos(k\pi h) \\ -\cos(k\pi h) & 1 \end{bmatrix} ,$$

by the Jacobi iteration in the frequency domain. In order to increase the convergence rate of (5), we have to reduce the spectral radius of $B(k)$.

4.2.2 Point SOR Iteration

The key idea of this paper is that instead of considering the SOR method for the large matrix corresponding to (1), we can study the SOR scheme for each small 2×2

matrix given by (6) separately and, then, seek the best SOR scheme for all of them. Once the SOR scheme is obtained in the frequency domain, we transform the problem back to the space domain so that the corresponding spatial SOR iteration can be determined.

It is important to observe in this context that $A(k)$ and $B(k)$ are *consistently ordered* matrix. Since the SOR method was originally developed to accelerate the convergence rate of consistently ordered matrices, the SOR method can be applied directly to the iteration (5). The definition of consistent ordering and the details of the SOR theory are all presented in [11] and [13]

Since this is a standard procedure, we only summarize the result here. Let

$$A(k) = I - L(k) - U(k)$$

where $L(k)$ and $U(k)$ are lower and upper triangular matrices respectively. Then, for a fixed frequency $k\pi$, the Jacobi iteration matrix is

$$B(k) = L(k) + U(k)$$

and the SOR iteration matrix associated with the frequency $k\pi$ is

$$G_\omega(k) = (I - \omega L(k))^{-1}\{(1-\omega)I + \omega U(k)\}. \tag{7}$$

In addition, the eigenvalues λ_k of $G_\omega(k)$ and the eigenvalues μ_k of $B(k)$ are related by [11, p.106]

$$(\lambda_k + \omega_k - 1)^2 = \lambda_k \omega_k^2 \mu_k^2 .$$

Hence,

$$\lambda_k = \left(\frac{\omega_k \mu_k \pm \sqrt{\Delta}}{2}\right)^2 \quad \text{where} \quad \Delta = \omega_k^2 \mu_k^2 - 4(\omega_k - 1),$$

and the spectral radius of (7) is

$$\rho_k = \begin{cases} \left(\dfrac{|\omega_k \mu_k| \pm \sqrt{\Delta}}{2} \right)^2 & \text{if } \Delta > 0 \\ \omega_k - 1 & \text{if } \Delta \leqslant 0 \end{cases} .$$

The above quantity can be minimized for all k by choosing

$$\omega^* = \frac{2}{1 + [1 - \mu_{\max}^2]^{1/2}} \quad \text{where} \quad \mu_{\max} = \max_{1 \leqslant k \leqslant N-1} |\mu_k| = \cos(\pi h) , \tag{8}$$

and the resulting spectral radius is

$$\rho^* = \omega^* - 1 \approx 1 - 2\sin(\pi h) = 1 - 2\pi h .$$

In particular, since the SOR method is applied to $A(k)$ partitioned with 1×1 diagonal submatrices, we call it the *point* SOR method.

The remaining problem is to transform the SOR iteration matrix (7) back to the space domain. By using the correspondence,

$$\cos(k\pi l h) = \frac{1}{2}(e^{ik\pi lh} + e^{-ik\pi lh}) \longleftrightarrow \frac{1}{2}(E^l + E^{-l}) \quad l = 1, 2, \cdots ,$$

where E^l and E^{-l} are the l-th order forward and backward shift operators defined as $E^l u_n = u_{n+l}$ and $E^{-l} u_n = u_{n-l}$, we find that the SOR iteration for r_n^m and b_n^m becomes

$$\begin{aligned} r_n^{m+1} &= (1 - \omega^*) r_n^m + \frac{\omega^*}{2}(b_{n-1}^m + b_{n+1}^m) \\ b_n^{m+1} &= (1 - \omega^*) b_n^m + \frac{\omega^*}{2}(r_{n-1}^{m+1} + r_{n+1}^{m+1}) \end{aligned} \tag{9}$$

It is straightforward to reconstruct the SOR iteration from (9), i.e.

$$u_n^{m+1} = (1 - \omega^*) u_n^m + \frac{\omega^*}{2}(u_{n-1}^m + u_{n+1}^m - h^2 f_n) \quad n \text{ even} ,$$

$$u_n^{m+1} = (1 - \omega^*) u_n^m + \frac{\omega^*}{2}(u_{n-1}^{m+1} + u_{n+1}^{m+1} - h^2 f_n) \quad n \text{ odd} ,$$

which is consistent with the conventional SOR method with red/black partitioning.

4.3 2-D 2-Level 4-color SOR Method

4.3.1 Problem Formulation

The 1-D 2-color SOR scheme discussed in the previous section can be naturally generalized to the 2-D case by using 4 colors.

Consider the following discretized system with uniform grid spacing h ,

$$\frac{1}{h^2}\{ q_1(u_{n_x+1,n_y}+u_{n_x-1,n_y})+q_2(u_{n_x,n_y+1}+u_{n_x,n_y-1})+q_3(u_{n_x+1,n_y+1}+u_{n_x+1,n_y-1}$$
$$+u_{n_x-1,n_y+1}+u_{n_x-1,n_y-1})-q\ u_{n_x,n_y}\}=f_{n_x,n_y} \qquad n_x,n_y=1,2,\cdots,\sqrt{N}-1, \quad (1)$$

where

$$q=2q_1+2q_2+4q_3,$$

and $\sqrt{N}=\frac{1}{h}$, and where q_1, q_2, and q_3 are nonnegative and not all zeros. It is also assumed that values at all boundary points are given. The system (1) can be viewed as obtained from a 5-point or 9-point stencil discretization of the equation

$$q'_1\frac{\partial^2 u(x,y)}{\partial x^2}+q'_2\frac{\partial^2 u(x,y)}{\partial y^2}=f(x,y) \quad \text{where } q'_1, q'_2>0, \qquad (2)$$

on the unit square $[0,1]^2$ with Dirichlet boundary conditions. In particular, when $q'_1=q'_2$, (2) becomes the Poisson equation. In this section, we present a frequency-domain approach for the design of a 2-level 4-color SOR method for the solution of (1). Several concrete examples will then be examined in Sections 4.4-4.6.

We can divide the grid points into four groups, say, red, black, green, and orange. A grid point is red if both n_x and n_y are even, black if n_x is odd and n_y is even, green if n_x is even and n_y is odd, and orange if both n_x and n_y are odd, as shown in Figure 4.1.

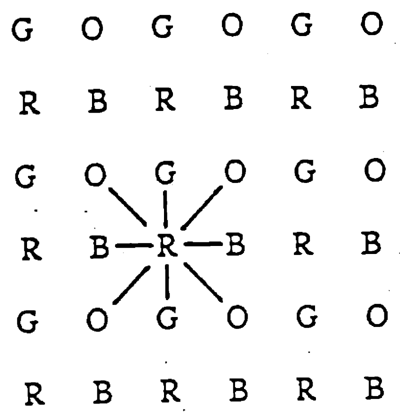


Figure 4.1: 4-color partitioning for the 9-point stencil discretization.

Following the procedure described in the previous section, to understand the error dynamics of the error associated to the Jacobi iteration for the system (1), we examine the dynamics of the four 2-D sequences

$$r^m_{n_x,n_y} = \sum_{k_x=1}^{\sqrt{N}-1} \sum_{k_y=1}^{\sqrt{N}-1} \hat{r}^m_{k_x,k_y} \sin(k_x \pi n_x h) \sin(k_y \pi n_y h) \quad 1 \leqslant n_x, n_y \leqslant \sqrt{N}-1,$$

$$b^m_{n_x,n_y} = \sum_{k_x=1}^{\sqrt{N}-1} \sum_{k_y=1}^{\sqrt{N}-1} \hat{b}^m_{k_x,k_y} \sin(k_x \pi n_x h) \sin(k_y \pi n_y h) \quad 1 \leqslant n_x, n_y \leqslant \sqrt{N}-1,$$

$$g^m_{n_x,n_y} = \sum_{k_x=1}^{\sqrt{N}-1} \sum_{k_y=1}^{\sqrt{N}-1} \hat{g}^m_{k_x,k_y} \sin(k_x \pi n_x h) \sin(k_y \pi n_y h) \quad 1 \leqslant n_x, n_y \leqslant \sqrt{N}-1,$$

$$o^m_{n_x,n_y} = \sum_{k_x=1}^{\sqrt{N}-1} \sum_{k_y=1}^{\sqrt{N}-1} \hat{o}^m_{k_x,k_y} \sin(k_x \pi n_x h) \sin(k_y \pi n_y h) \quad 1 \leqslant n_x, n_y \leqslant \sqrt{N}-1,$$

where the coefficients $\hat{r}^m_{k_x,k_y}$, $\hat{b}^m_{k_x,k_y}$, $\hat{g}^m_{k_x,k_y}$, $\hat{o}^m_{k_x,k_y}$ are chosen such that

$$r^m_{n_x,n_y} = e^m_{n_x,n_y} \quad n_x \text{ even } n_y \text{ even}, \qquad b^m_{n_x,n_y} = e^m_{n_x,n_y} \quad n_x \text{ odd } n_y \text{ even},$$

$$g^m_{n_x,n_y} = e^m_{n_x,n_y} \quad n_x \text{ even } n_y \text{ odd}, \qquad o^m_{n_x,n_y} = e^m_{n_x,n_y} \quad n_x \text{ odd } n_y \text{ odd},$$

where $e^m_{n_x,n_y} = u^m_{n_x,n_y} - \bar{u}_{n_x,n_y}$ is the m th iteration error at grid point (n_x, n_y).

As shown before, we can transform the Jacobi iteration for $r^m_{n_x,n_y}$, $b^m_{n_x,n_y}$, $g^m_{n_x,n_y}$ and $o^m_{n_x,n_y}$, or equivalently for the errors in the space domain at the red, black, green, and orange points, into an equivalent set of iterations for the Fourier coefficients $\hat{r}^m_{k_x,k_y}$, $\hat{b}^m_{k_x,k_y}$, $\hat{g}^m_{k_x,k_y}$ and $\hat{o}^m_{k_x,k_y}$ in the frequency domain. These iterations can be viewed as solving the system

$$A(k_x,k_y) \begin{bmatrix} \hat{r}^\infty_{k_x,k_y} \\ \hat{o}^\infty_{k_x,k_y} \\ \hat{b}^\infty_{k_x,k_y} \\ \hat{g}^\infty_{k_x,k_y} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix},$$

where

$$A(k_x,k_y) = \begin{bmatrix} 1 & -\alpha_3 & -\alpha_1 & -\alpha_2 \\ -\alpha_3 & 1 & -\alpha_2 & -\alpha_1 \\ -\alpha_1 & -\alpha_2 & 1 & -\alpha_3 \\ -\alpha_2 & -\alpha_1 & -\alpha_3 & 1 \end{bmatrix}, \qquad (3)$$

and where

$$\alpha_1 = \frac{2q_1\cos(k_x \pi h)}{q}, \alpha_2 = \frac{2q_2\cos(k_y \pi h)}{q}, \alpha_3 = \frac{4q_3\cos(k_x \pi h)\cos(k_y \pi h)}{q}. \qquad (4)$$

4.3.2 Block SOR Iteration

The matrix $A(k_x,k_y)$ partitioned such that its diagonal submatrices are all 1×1 matrices is *not* a consistently ordered matrix. However, if $A(k_x,k_y)$ is partitioned with 2×2 block diagonal submatrices, it is a block consistently ordered matrix. Hence, the *block* SOR iteration can be applied to $A(k_x,k_y)$ with this kind of partitioning.

The matrix A can be written as

$$A(k_x,k_y) = D(k_x,k_y) - E(k_x,k_y) - F(k_x,k_y)$$

where

$$D(k_x,k_y) = \begin{bmatrix} 1 & -\alpha_3 & 0 & 0 \\ -\alpha_3 & 1 & 0 & 0 \\ 0 & 0 & 1 & -\alpha_3 \\ 0 & 0 & -\alpha_3 & 1 \end{bmatrix}, \quad E(k_x,k_y) = \begin{bmatrix} 0 & 0 & \alpha_1 & \alpha_2 \\ 0 & 0 & \alpha_2 & \alpha_1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

and $F(k_x,k_y) = E^T(k_x,k_y)$. In addition, we can define $L(k_x,k_y) \equiv D^{-1}(k_x,k_y)E(k_x,k_y)$ and $U(k_x,k_y) \equiv D^{-1}(k_x,k_y)F(k_x,k_y)$, i.e.,

$$L(k_x,k_y) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \beta_1 & \beta_2 & 0 & 0 \\ \beta_2 & \beta_1 & 0 & 0 \end{bmatrix}, \quad U(k_x,k_y) = \begin{bmatrix} 0 & 0 & \beta_1 & \beta_2 \\ 0 & 0 & \beta_2 & \beta_1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix},$$

where

$$\beta_1 = \frac{\alpha_1 + \alpha_2\alpha_3}{1-\alpha_3^2}, \qquad \beta_2 = \frac{\alpha_2 + \alpha_1\alpha_3}{1-\alpha_3^2}.$$

Then, the block Jacobi iteration matrix is

$$B(k_x, k_y) = L(k_x, k_y) + U(k_x, k_y), \tag{5}$$

and the corresponding block SOR iteration matrix is

$$G_\omega(k_x, k_y) = (I - \omega L(k_x, k_y))^{-1}\{(1-\omega)I + \omega U(k_x, k_y)\}. \tag{6}$$

It is easy to find that the eigenvalues of $B(k_x, k_y)$ are double roots at

$$\mu_{k_x, k_y} = \beta_1 \pm \beta_2 = \frac{\alpha_1+\alpha_2}{1-\alpha_3}, \frac{\alpha_1-\alpha_2}{1+\alpha_3}. \tag{7}$$

In addition, the eigenvalues μ_{k_x, k_y} of the Jacobi iteration matrix $B(k_x, k_y)$ and the eigenvalues λ_{k_x, k_y} of the SOR iteration matrix $G_\omega(k_x, k_y)$ are related by

$$(\lambda_{k_x, k_y} + \omega_{k_x, k_y} - 1)^2 = \lambda_{k_x, k_y}\, \omega_{k_x, k_y}^2\, \mu_{k_x, k_y}^2.$$

Hence, if we proceed as in the 1-D case, except for a change of subscript from the 1-D index ξ to the 2-D index (k_x, k_y), we find that

$$\lambda_{k_x, k_y} = \left(\frac{\omega_{k_x, k_y}\mu_{k_x, k_y} \pm \sqrt{\Delta}}{2}\right)^2 \quad \text{where} \quad \Delta = \omega_{k_x, k_y}^2 \mu_{k_x, k_y}^2 - 4(\omega_{k_x, k_y} - 1)$$

and the spectral radius of $G_\omega(k_x, k_y)$ is

$$\rho_{k_x, k_y} = \begin{cases} \left(\dfrac{|\omega_{k_x, k_y}\mu_{k_x, k_y}| \pm \sqrt{\Delta}}{2}\right)^2 & \text{if } \Delta > 0 \\ \omega_{k_x, k_y} - 1 & \text{if } \Delta \leqslant 0 \end{cases}.$$

The above quantity is minimized for all ξ and η by choosing the following optimal relaxation parameter

$$\omega^* = \frac{2}{1 + [1-\mu_{\max}^2]^{1/2}} \quad \text{where} \quad \mu_{\max} = \max_{1 \leqslant k_x, k_y \leqslant \sqrt{N}-1} |\mu_{k_x, k_y}|, \tag{8}$$

and the spectral radius of the corresponding SOR matrix is

$$\rho^* = \omega^* - 1 \ .$$

4.3.3 2-level SOR Iteration

Suppose that one of the coefficients q_1, q_2 or q_3 is zero, or equivalently, that one of α_1, α_2, or α_3 is zero for all (k_x, k_y). Then, the 4-color block SOR method described above reduces to an equivalent 2-color SOR method, which corresponds to a degenerate case that will be discussed in Section 4.4.

For the moment, consider the nondegenerate case in which q_1, q_2, and q_3 are all greater than zero. In this case, the *pure* frequency-domain block SOR method given by (6) cannot be successfully transformed back to the space domain. To see this, let us write the space domain equation corresponding to (6),

$$\begin{aligned}
r^{m+1}_{n_x,n_y} &= (1-\omega^*)\, r^m_{n_x,n_y} + \omega^* (\beta_1^S b^m_{n_x,n_y} + \beta_2^S g^m_{n_x,n_y}), \\
o^{m+1}_{n_x,n_y} &= (1-\omega^*)\, o^m_{n_x,n_y} + \omega^* (\beta_2^S b^m_{n_x,n_y} + \beta_1^S g^m_{n_x,n_y}), \\
b^{m+1}_{n_x,n_y} &= (1-\omega^*)\, b^m_{n_x,n_y} + \omega^* (\beta_1^S r^{m+1}_{n_x,n_y} + \beta_2^S o^{m+1}_{n_x,n_y}), \\
g^{m+1}_{n_x,n_y} &= (1-\omega^*)\, g^m_{n_x,n_y} + \omega^* (\beta_2^S r^{m+1}_{n_x,n_y} + \beta_1^S o^{m+1}_{n_x,n_y}),
\end{aligned} \qquad (9)$$

where β_1^S and β_2^S represent the space domain operators corresponding to β_1 and β_2 in the frequency domain. It is straightforward to derive that

$$\beta_1^S = \frac{qq_1(E_1+E_1^{-1}) + q_2q_3(E_1+E_1^{-1})^2(E_2+E_2^{-1})}{q^2 - q_3^2(E_1+E_1^{-1})^2(E_2+E_2^{-1})^2}$$

$$\approx \{ qq_1(E_1+E_1^{-1}) + q_2q_3(E_1+E_1^{-1})^2(E_2+E_2^{-1}) \} \times$$

$$\frac{1}{q^2} \{ 1 + (\frac{q_3}{q})^2(E_1+E_1^{-1})^2(E_2+E_2^{-1})^2 + \cdots \} ,$$

where E_1, E_1^{-1}, E_2, and E_2^{-1} are forward and backward shift operators in the n_x and n_y directions respectively. A similar expression can be written for β_2^S by interchanging q_1 and q_2. Although values of the solution corresponding to different colors at the $n+1$th iteration are totally decoupled in (9), the implementation of (9) is very expensive in terms of the amount of computation and communication required.

In order to avoid this difficulty, we can rewrite (6) as

$$G_\omega(k_x, k_y) = (D(k_x, k_y) - \omega E(k_x, k_y))^{-1} \{ (1-\omega) D(k_x, k_y) + \omega F(k_x, k_y) \},$$

and the corresponding space domain equation associated to the optimal block relaxation parameter ω^* becomes

$$\begin{aligned}
r_{n_x,n_y}^{m+1} - \alpha_3^S o_{n_x,n_y}^{m+1} &= (1-\omega^*)(r_{n_x,n_y}^m - \alpha_3^S o_{n_x,n_y}^m) + \omega^* (\alpha_1^S b_{n_x,n_y}^m + \alpha_2^S g_{n_x,n_y}^m), \\
-\alpha_3^S r_{n_x,n_y}^{m+1} + o_{n_x,n_y}^{m+1} &= (1-\omega^*)(-\alpha_3^S r_{n_x,n_y}^m + o_{n_x,n_y}^m) + \omega^* (\alpha_2^S b_{n_x,n_y}^m + \alpha_1^S g_{n_x,n_y}^m), \\
b_{n_x,n_y}^{m+1} - \alpha_3^S g_{n_x,n_y}^{m+1} &= (1-\omega^*)(b_{n_x,n_y}^m - \alpha_3^S g_{n_x,n_y}^m) + \omega^* (\alpha_1^S r_{n_x,n_y}^{m+1} + \alpha_2^S o_{n_x,n_y}^{m+1}), \\
-\alpha_3^S b_{n_x,n_y}^{m+1} + g_{n_x,n_y}^{m+1} &= (1-\omega^*)(-\alpha_3^S b_{n_x,n_y}^m + g_{n_x,n_y}^m) + \omega^* (\alpha_2^S r_{n_x,n_y}^{m+1} + \alpha_1^S o_{n_x,n_y}^{m+1}),
\end{aligned} \tag{10}$$

where α_1^S, α_2^S, and α_3^S are space domain operators corresponding respectively to α_1, α_2, and α_3 respectively, i.e.

$$\alpha_1^S = \frac{q_1}{q}(E_1 + E_1^{-1}), \qquad \alpha_2^S = \frac{q_2}{q}(E_2 + E_2^{-1}),$$

$$\alpha_3^S = \frac{q_3}{q}(E_1 + E_1^{-1})(E_2 + E_2^{-1}).$$

The right-hand side of (10) is much easier to implement than that of (9). However, the price that we pay is that values at the red and orange points and values of the black and green points at the $(m+1)$th iteration are coupled together as indicated in the left-hand side of (10)

We can divide (10) into two sets of equations: the first two and the last two. Within each set, for example the first two equations, instead of solving r_{n_x,n_y}^{m+1} and o_{n_x,n_y}^{m+1} directly, we can apply a point SOR scheme to these two equations and solve

r_{n_x,n_y}^{m+1} and o_{n_x,n_y}^{m+1} iteratively. As a consequence, we obtain a 2-level SOR method.

A precise description of the 2-level SOR algorithm for (1) in terms of the matrix $A(k_x,k_y)$ specified by (3) is given in Table 4.1. At the 1st level, we treat all red and orange points as a group (n_x+n_y even) and all black and green points as another group (n_x+n_y odd) and perform a block SOR iteration between these two groups. The output of this iteration is denoted by $ff_{n_x,n_y}^{m_b+1}$ and is used as driving function for the 2nd-level iteration. At the 2nd level, we perform a point SOR iteration to further decouple values at red and orange points, or at black and green points.

A data flow diagram which illustrates how grid points exchange values with their neighboring points at each iteration for the above algorithm is shown in Figure 4.2.

Note that we use different relaxation parameters at different levels, i.e., we use respectively ω_b and ω_p for the block and point SOR iterations. It is a well known result that both the block and point SOR iterations applied to a positive definite matrix converge if and only if their relaxation parameters are between 0 and 2 [11]. Hence, the convergence of the 2-level SOR scheme that we present can be achieved by first choosing

$$0 < \omega_p < 2 , \quad M \text{ sufficiently large} , \tag{11a}$$

where M denotes the total number of point SOR iterations performed at the second level, so that the point SOR iteration converges inside each block SOR iteration. Under condition (11a), a 2-level SOR iteration is not different from a single-level block SOR iteration. Therefore, by imposing the additional constraint,

$$0 < \omega_b < 2 , \tag{11b}$$

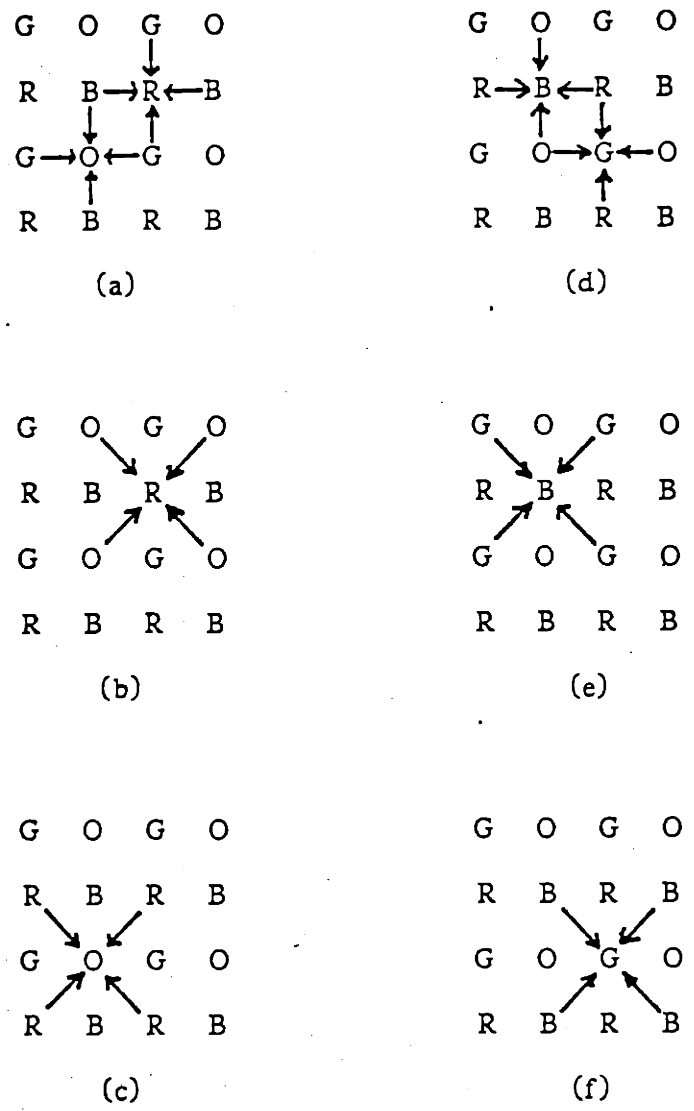


Figure 4.2: Data flow diagram for a 2-level 4-color SOR method with computational order { red → orange → black → green } (a) Block SOR iteration for red and orange points. (b) and (c) Point SOR iterations for red and orange points. (d) Block SOR iteration for black and green points. (e) and (f) Point SOR iterations for black and green points.

```
for $m_b = 0, 1, 2, \cdots$
    /beginning of a block SOR sweep/
    for $n_x + n_y$ even    /red & orange/
    $ff_{n_x,n_y}^{m_b+1} := (1-\omega_b)(u_{n_x,n_y}^{m_b} - \alpha_3^S u_{n_x,n_y}^{m_b}) + \omega_b(\alpha_1^S u_{n_x,n_y}^{m_b} + \alpha_2^S u_{n_x,n_y}^{m_b} - h^2 f_{n_x,n_y})$
    $v_{n_x,n_y}^0 := u_{n_x,n_y}^{m_b}$
        for $m_p = 0, 1, 2, \dots, M-1$
            /beginning of a point SOR sweep at red & orange points/
            for $n_y$ even    /red/
            $v_{n_x,n_y}^{m_p+1} := (1-\omega_p)v_{n_x,n_y}^{m_p} + \omega_p(\alpha_3^S v_{n_x,n_y}^{m_p} + ff_{n_x,n_y}^{m_b+1})$
            for $n_y$ odd    /orange/
            $v_{n_x,n_y}^{m_p+1} := (1-\omega_p)v_{n_x,n_y}^{m_p} + \omega_p(\alpha_3^S v_{n_x,n_y}^{m_p+1} + ff_{n_x,n_y}^{m_b+1})$
            /end of the point SOR sweep at red & orange points/
    $u_{n_x,n_y}^{m_b+1} := v_{n_x,n_y}^M$
    for $n_x + n_y$ odd    /black & green/
    $ff_{n_x,n_y}^{m_b+1} := (1-\omega_b)(u_{n_x,n_y}^{m_b} - \alpha_3^S u_{n_x,n_y}^{m_b}) + \omega_b(\alpha_1^S u_{n_x,n_y}^{m_b+1} + \alpha_2^S u_{n_x,n_y}^{m_b+1} - h^2 f_{n_x,n_y})$
    $v_{n_x,n_y}^0 := u_{n_x,n_y}^{m_b}$
        for $m_p = 0, 1, 2, \dots, M-1$
            /beginning of a point SOR sweep at black & green points/
            for $n_y$ even    /black/
            $v_{n_x,n_y}^{m_p+1} := (1-\omega_p)v_{n_x,n_y}^{m_p} + \omega_p(\alpha_3^S v_{n_x,n_y}^{m_p} + ff_{n_x,n_y}^{m_b+1})$
            for $n_y$ odd    /green/
            $v_{n_x,n_y}^{m_p+1} := (1-\omega_p)v_{n_x,n_y}^{m_p} + \omega_p(\alpha_3^S v_{n_x,n_y}^{m_p+1} + ff_{n_x,n_y}^{m_b+1})$
            /end of the point SOR sweep at black & green points/
    $u_{n_x,n_y}^{m_b+1} := v_{n_x,n_y}^M$
    /end of the block SOR sweep/
```

Table 4.1: 2-level 4-color SOR Method

the 2-level SOR method is guaranteed to converge. In Section 4.5, we will discuss how to select the value of M and optimal relaxation parameters ω_p^* and ω_b^* to maximize the convergence rate of the 2-level SOR method.

4.4 Degenerate Case: 5-point Stencils

In this section, we show that the traditional single-level 2-color SOR method for a 5-point stencil is in fact a degenerate case of the general 2-level 4-color SOR method described in Table 1. The following discussion also gives us more insight into the 2-level SOR algorithm.

4.4.1 Standard 5-point Stencil

The standard 5-point stencil discretization of the Poisson equation is

$$\frac{1}{h^2}\left(u_{n_x+1,n_y} + u_{n_x-1,n_y} + u_{n_x,n_y+1} + u_{n_x,n_y-1} - 4\, u_{n_x,n_y} \right) = f_{n_x,n_y} ,$$

which is a special case of (3.1) with

$$q_1 = 1 , \quad q_2 = 1 , \quad q_3 = 0 , \quad \text{and} \quad q = 4 .$$

Hence, we have

$$\alpha_1 = \frac{\cos(k_x \pi h)}{2} , \quad \alpha_2 = \frac{\cos(k_y \pi h)}{2} , \quad \alpha_3 = 0 ,$$

and

$$\alpha_1^S = \frac{E_x + E_x^{-1}}{4} , \quad \alpha_2^S = \frac{E_y + E_y^{-1}}{4} , \quad \alpha_3^S = 0 .$$

For this case, we know that $\omega_p^* = 0$ from (2.8). It is easy to check that the 2nd-level point SOR iteration becomes trivial and that only the 1st-level block SOR iteration is necessary, which is exactly the same as the traditional red/black SOR method with the following optimal relaxation parameter

$$\omega^* = \omega_b^* = \frac{2}{1 + [1-\cos^2(\pi h)]^{1/2}} \approx 2 - 2\pi h . \qquad (1)$$

4.4.2 Rotated 5-point Stencil

Another 5-point stencil discretization of the Poisson equation is [5]

$$\frac{1}{2h^2}\left(u_{n_x+1,n_y+1} + u_{n_x+1,n_y-1} + u_{n_x-1,n_y+1} + u_{n_x-1,n_y-1} - 4\, u_{n_x,n_y} \right) = f_{n_x,n_y} ,$$

which is also a special case of (3.1) with

$$q_1 = 0 , \quad q_2 = 0 , \quad q_3 = \frac{1}{2} , \quad \text{and} \quad q = 2 .$$

Consequently, we find that

$$\alpha_1 = 0 , \quad \alpha_2 = 0 , \quad \alpha_3 = \cos(k_x \pi h)\cos(k_y \pi h) ,$$

and

$$\alpha_1^S = 0 , \quad \alpha_2^S = 0 , \quad \alpha_3^S = \frac{(E_x + E_x^{-1})(E_y + E_y^{-1})}{4} .$$

It turns out that $\omega_b^* = 1$ and in this case the 1-st level block SOR iteration becomes trivial. Only the 2nd-level point SOR iteration is necessary, which can be written as

$$u_{n_x,n_y}^{m+1} = (1-\omega^*)\, u_{n_x,n_y}^{m} + \omega^*(\alpha_3^S u_{n_x,n_y}^{m} - \tfrac{1}{2}h^2 f_{n_x,n_y}) \qquad (n_x,n_y) \text{ red } \textit{or} \text{ black}$$

$$u_{n_x,n_y}^{m+1} = (1-\omega^*)\, u_{n_x,n_y}^{m} + \omega^*(\alpha_3^S u_{n_x,n_y}^{m+1} - \tfrac{1}{2}h^2 f_{n_x,n_y}) \qquad (n_x,n_y) \text{ orange } \textit{or} \text{ green}$$

where

$$\omega^* = \omega_p^* = \frac{2}{1 + [1-\cos^4(\pi h)]^{1/2}} \approx 2 - 2\sqrt{2}\pi h . \qquad (2)$$

By comparing (1) and (2), we find that the only difference between the standard and rotated 5-point stencil discretizations is that the mesh size is h in the first case, and $\sqrt{2}h$ in the second case. The optimal relaxation parameter ω^* and spectral radius $\rho^* = \omega^* - 1$ have therefore to be adjusted accordingly. Note however that the above observation depends on the isotropy of the Poisson equation, since the standard and rotated 5-point stencils give rise to different discretizations in the anisotropic case.

4.5 Convergence Rate Analysis

In this section, we show how to select the optimal relaxation parameters ω_b^* and ω_p^* for the 2-level 4-color SOR method described in Section 4.3, and we analyze the convergence rate of the resulting method when it is applied to equation (4.3.1) with nondegenerate coefficients, i.e., for

$$q_1 > 0, \quad q_2 > 0, \quad q_3 > 0 .$$

4.5.1 Determination of Optimal 2-level Relaxation Parameters

First, let us concentrate on the 2nd-level point iteration. In order to determine the optimal relaxation parameter, we need to find the spectral radius of the point Jacobi iteration which is given by

$$\mu_{p,max} = \max_{1 \le k_x, k_y \le \sqrt{N}-1} |\alpha_3| = \frac{4q_3 \cos^2(\pi h)}{q} \approx \frac{4q_3}{q}(1-\pi^2 h^2) ,$$

where the maximum value of $|\alpha_3|$ occurs for $(k_x, k_y) = (1,1)$ and $(\sqrt{N}-1, \sqrt{N}-1)$. Since the spectral radius of the point Jacobi iteration is bounded by the constant $\frac{4q_3}{q}$ which is less than 1, even a simple point Jacobi relaxation converges reasonably fast. Nevertheless, this can be further improved by a point SOR iteration using the following optimal relaxation parameter

$$\omega_p^* = \frac{2}{1 + [1 - \frac{(4q_3)^2 \cos^4(\pi h)}{q^2}]^{1/2}} \approx 1 + \frac{1}{4}(\frac{4q_3}{q})^2 , \tag{1}$$

with the spectral radius

$$\rho_p^* = \omega_p^* - 1 \approx \frac{1}{4}(\frac{4q_3}{q})^2 . \tag{2}$$

For a typical example, we have $q_3 = 1$ and $q = 20$ (see Section 4.6) so that $\rho_p^* \approx 0.01$.

Since the error can be damped approximately at the rate 10^{-2M}, where M is the number of 2nd-level iterations, only 2 or 3 point SOR iterations inside each block SOR iteration are necessary. The fact that the 2nd-level point SOR iteration requires only a constant number M of steps to converge, where M is usually 2 or 3, plays a crucial role in our analysis of the convergence rate of the 2-level SOR method. By using this observation, it will be shown below that the convergence rate of the 2-level SOR scheme is similar to that of the standard SOR method for a 5-point stencil, or of the 9-point SOR scheme discussed in [2].

Next, we examine the 1st-level block iteration. The spectral radius of the block Jacobi iteration matrix (4.3.5) is given by

$$\mu_{b,max} = \max_{1 \leq k_x, k_y \leq \sqrt{N}-1} \{ \left| \frac{\alpha_1+\alpha_2}{1-\alpha_3} \right| , \left| \frac{\alpha_1-\alpha_2}{1+\alpha_3} \right| \} = \frac{2(q_1+q_2)\cos(\pi h)}{q-4q_3\cos^2(\pi h)} ,$$

which occurs at

$$(k_x, k_y) = (1,1), (1,\sqrt{N}-1), (\sqrt{N}-1,1) \text{ and } (\sqrt{N}-1,\sqrt{N}-1) .$$

By using the fact that $q=2q_1+2q_2+4q_4$, we can simplify $\mu_{b,max}$ as

$$\mu_{b,max} = \frac{(q-4q_3)\cos(\pi h)}{q-4q_3\cos^2(\pi h)} \approx 1 - (\frac{1}{2}+\frac{4q_3}{q-4q_3})\pi^2 h^2 .$$

Hence, the optimal relaxation parameter for the block SOR iteration is

$$\omega_b^* = \frac{2}{1+(1-\mu_{b,max}^2)^{1/2}} \approx 2 - 2\,(1+\frac{8q_3}{q-4q_3})^{1/2}\pi h \ , \tag{3}$$

and the spectral radius is

$$\rho_b^* = \omega_b^* - 1 \approx 1 - 2\,(1+\frac{8q_3}{q-4q_3})^{1/2}\pi h \ . \tag{4}$$

Therefore, if $q_3 = 1$ and $q = 20$, then $\rho_b^* = 1 - \sqrt{6}\pi h$.

Since for a fixed point, the 2-level SOR method divides neighboring points into two groups and operates on one group at the block iteration level and on the other group at the point iteration level, and since each block SOR iteration at the first level requires M point SOR iterations at the second level, it is convenient to define the *effective* number of iterations for one 2-level SOR iteration as

$$n_{eff} \equiv \frac{w_p M + w_b}{w_b + w_p} , \tag{5}$$

where w_b and w_p represent the amount of work required per block and per point iteration respectively. The number n_{eff} measures approximately the computational burden of one full 2-level SOR iteration in terms of equivalent 9-point Jacobi iterations.

If the point SOR iteration converges in M iterations, the convergence rate of the 2-level SOR method is then only determined by that of the block SOR iteration. Therefore, we can define the *effective* spectral radius of the 2-level SOR iteration as

$$\rho_{eff}^* \equiv (\rho_b^*)^{\frac{1}{n_{eff}}} = (\rho_b^*)^{\frac{w_b + w_p}{w_p M + w_b}} , \tag{6}$$

which is used to measure the average smoothing rate per effective iteration of the 2-level SOR scheme.

For the above example, since the amount of computational work for each block and point SOR iteration is the same, we have $w_p = w_b$, so that

$$n_{eff} = \frac{M+1}{2} , \quad \rho_{eff}^* \approx 1 - \frac{2}{M+1} \sqrt{6} \pi h .$$

When $M = 2$, we find therefore that

$$n_{eff} = \frac{3}{2} = 1.5 , \quad \rho_{eff}^* \approx 1 - 1.63 \pi h . \tag{7}$$

The above effective spectral radius ρ_{eff}^* should be compared with the spectral radius $\rho_9^* \approx 1 - 1.79\pi h$ that was obtained for the 9-point SOR method discussed by [2]. In the next section, we will present a 2-level SOR method with a different computational ordering whose effective spectral radius is $\rho_{eff}^* \approx 1 - 2.26\pi h$.

We see from the above comparison that the 2-level SOR method and the 9-point SOR procedure of [2] have very similar convergence rates. The main difference is of course that the method of [2] is a single-level method which uses only one relaxation parameter ω^* . In addition, its convergence rate analysis requires the study of the solution of a quartic equation, and does not yield closed-form relations between ρ^* , ω^* and the spectral radius μ of the 9-point Jacobi iteration matrix. By comparison, the approach that we have used above to study the convergence of the 2-level SOR method relies on standard SOR theory, and provides closed-form relations between ρ_p^*, ω_p^*, and $\mu_{p,max}$, and between ρ_b^*, ω_b^*, and $\mu_{b,max}$.

Finally, note that the amount of work required by each effective iteration for the 9-point stencil case is about twice as large as for a standard 5-point SOR iteration. Thus, to compare the convergence rate of the 2-level SOR method with that of the standard 5-point SOR scheme, we must compare ρ_{eff}^* with the spectral radius $(\rho_5^*)^2 \approx 1 - 4\pi h$ corresponding to two 5-point SOR iterations. This comparison seems to indicate that the 5-point SOR iteration converges faster than the 2-level SOR method, or the 9-point SOR method discussed in [2]. However, the 9-point stencil discretization is more accurate than the corresponding 5-point stencil discretization. Thus, for the same accuracy, we can select h larger for the 9-point stencil discretization so that in actuality the 2-level or single-level 9-point SOR methods may converge faster than the standard 5-point SOR method.

4.5.2 Computational Order

In the above discussion, we have used a particular computational order, i.e., { red $\rightarrow$ orange $\rightarrow$ black $\rightarrow$ green }. Now, let us consider other computational orderings. Although there exist $4! = 24$ different ways to permute the computational order for these 4 colors, they only result in 3 different 2-level SOR iteration schemes. By interchanging the relative positions of α_1, α_2, and α_3 in the matrix $A(k_x, k_y)$, we can obtain only 6 different matrices, each of which corresponds to 4 different computational orderings. Furthermore, we can divide these 6 matrices into 3 classes:

$$\text{Class 1}: \begin{bmatrix} 1 & -\alpha_1 & -\alpha_2 & -\alpha_3 \\ -\alpha_1 & 1 & -\alpha_3 & -\alpha_2 \\ -\alpha_2 & -\alpha_3 & 1 & -\alpha_1 \\ -\alpha_3 & -\alpha_2 & -\alpha_1 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & -\alpha_1 & -\alpha_3 & -\alpha_2 \\ -\alpha_1 & 1 & -\alpha_2 & -\alpha_3 \\ -\alpha_3 & -\alpha_2 & 1 & -\alpha_1 \\ -\alpha_2 & -\alpha_3 & -\alpha_1 & 1 \end{bmatrix}$$

$$\text{Class 2}: \begin{bmatrix} 1 & -\alpha_2 & -\alpha_1 & -\alpha_3 \\ -\alpha_2 & 1 & -\alpha_3 & -\alpha_1 \\ -\alpha_1 & -\alpha_3 & 1 & -\alpha_2 \\ -\alpha_3 & -\alpha_1 & -\alpha_2 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & -\alpha_2 & -\alpha_3 & -\alpha_1 \\ -\alpha_2 & 1 & -\alpha_1 & -\alpha_3 \\ -\alpha_3 & -\alpha_1 & 1 & -\alpha_2 \\ -\alpha_1 & -\alpha_3 & -\alpha_2 & 1 \end{bmatrix}$$

$$\text{Class 3}: \begin{bmatrix} 1 & -\alpha_3 & -\alpha_1 & -\alpha_2 \\ -\alpha_3 & 1 & -\alpha_2 & -\alpha_1 \\ -\alpha_1 & -\alpha_2 & 1 & -\alpha_3 \\ -\alpha_2 & -\alpha_1 & -\alpha_3 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & -\alpha_3 & -\alpha_2 & -\alpha_1 \\ -\alpha_3 & 1 & -\alpha_1 & -\alpha_2 \\ -\alpha_2 & -\alpha_1 & 1 & -\alpha_3 \\ -\alpha_1 & -\alpha_2 & -\alpha_3 & 1 \end{bmatrix}$$

It is easy to see that the same 2-level SOR method applies to matrices within the same class. Although the discussion in Section 4.1 applies only to matrices of Class 3, we can use a similar approach to obtain optimal block and point relaxation parameters and spectral radii for a 2-level SOR method for matrices of Classes 1 and 2. For matrices of Class 1, we find

$$\omega_p^* \approx 1 + \frac{1}{4}(\frac{2q_1}{q})^2 , \quad \rho_p^* \approx \frac{1}{4}(\frac{2q_1}{q})^2 , \tag{8}$$

$$\omega_b^* \approx 2 - 2\,(\frac{q+4q_3}{q-2q_1})^{1/2}\pi h , \quad \rho_b^* \approx 1 - 2\,(\frac{q+4q_3}{q-2q_1})^{1/2}\pi h . \tag{9}$$

and for matrices of Class 2, we need only to replace q_1 by q_2 in the above expressions.

The data flow diagram for the computational order { red $\rightarrow$ black $\rightarrow$ green $\rightarrow$ orange }, which corresponds to a 2-level SOR method applied to matrices of Class 1, is shown in Figure 4.3. Let us analyze the convergence rate for this 2-level SOR iteration. From Figure 4.3, it is easy to see that $w_p = \frac{1}{3} w_b$. Therefore, from (5) and (6), we have

$$n_{eff} = \frac{3+M}{4} , \quad \rho_{eff}^* = (\rho_b^*)^{\frac{4}{3+M}} .$$

Consider now the typical example where $q_1 = q_2 = 4$ and $q_3 = 1$. By using (8) and (9), we find that the spectral radius of the point SOR iteration becomes larger, but the spectral radius of the block SOR iteration becomes smaller, i.e.

$$\rho_p^* \approx 4 \times 10^{-2} , \quad \rho_b^* \approx 1 - \sqrt{8}\pi h .$$

Therefore, the effective spectral radius can be expressed as

$$\rho_{eff}^* \approx 1 - \frac{4}{M+3}\sqrt{8}\pi h .$$

This gives

$$\rho_{eff}^* \approx 1 - 2.26\pi h \quad \text{if } M = 2, \qquad \rho_{eff}^* \approx 1 - 1.89\pi h \quad \text{if } M = 3. \tag{10}$$

By comparing (7) and (10), we observe that the performance of a 2-level SOR iteration applied to matrices of the first or second class is in fact better for this specific example.

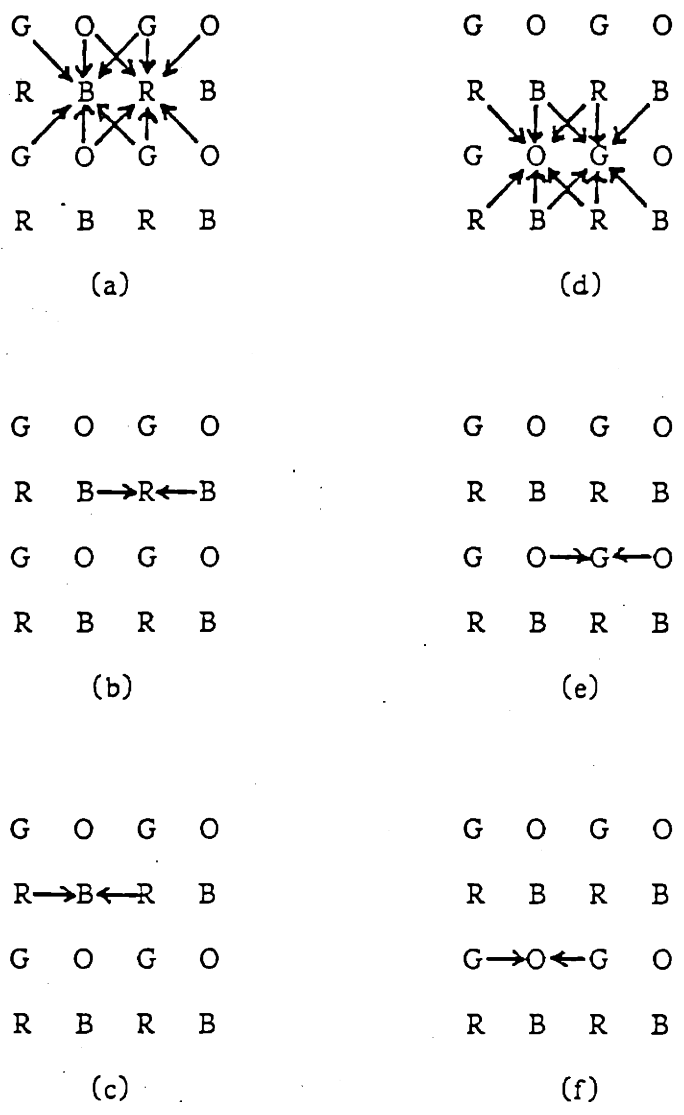


Figure 4.3: Data flow diagram for a 2-level 4-color SOR method with computational order { red $\rightarrow$ black $\rightarrow$ green $\rightarrow$ orange }. (a) Block SOR iteration for red and black points. (b) and (c) Point SOR iterations for red and black points. (d) Block SOR iteration for orange and green points. (e) and (f) Point SOR iterations for orange and green points.

4.6 Numerical Examples

We consider the system of equations obtained from a 9-point stencil discretization of the isotropic Poisson equation, i.e.,

$$\frac{1}{6h^2}\{\, 4(u_{n_x+1,n_y}+u_{n_x-1,n_y})+4(u_{n_x,n_y+1}+u_{n_x,n_y-1})+(u_{n_x+1,n_y+1}+u_{n_x+1,n_y-1}$$
$$+u_{n_x-1,n_y+1}+u_{n_x-1,n_y-1})-20\,u_{n_x,n_y}\,\}=f_{n_x,n_y}\qquad n_x,n_y=1,2,\cdots,\sqrt{N}-1, \qquad (1)$$

with zero boundary conditions and $h=\frac{1}{\sqrt{N}}=\frac{1}{20}$. In this case, $q_1=q_2=4$, and $q_3=1$. Since in this example the performance of the 2-level SOR method for matrices $A(k_x,k_y)$ of Classes 1 and 2 is the same, we compare only the following two computational orders:

order (a): { red $\rightarrow$ orange $\rightarrow$ black $\rightarrow$ green },

order (b): { red $\rightarrow$ black $\rightarrow$ green $\rightarrow$ orange }.

The computational orders (a) and (b) are obtained by applying the 2-level SOR iteration to matrices $A(k_x,k_y)$ belonging respectively to Classes 3 and 1. Their spectral radii and optimal relaxation parameters for the block SOR and the point SOR iterations are summarized in Table 4.2.

order	ω_b^*	ρ_b^*	ω_p^*	ρ_p^*
(a)	1.679931	0.679931	1.009702	0.009702
(b)	1.640105	0.640105	1.042400	0.042400

Table 4.2: Optimal block and point relxation parameters

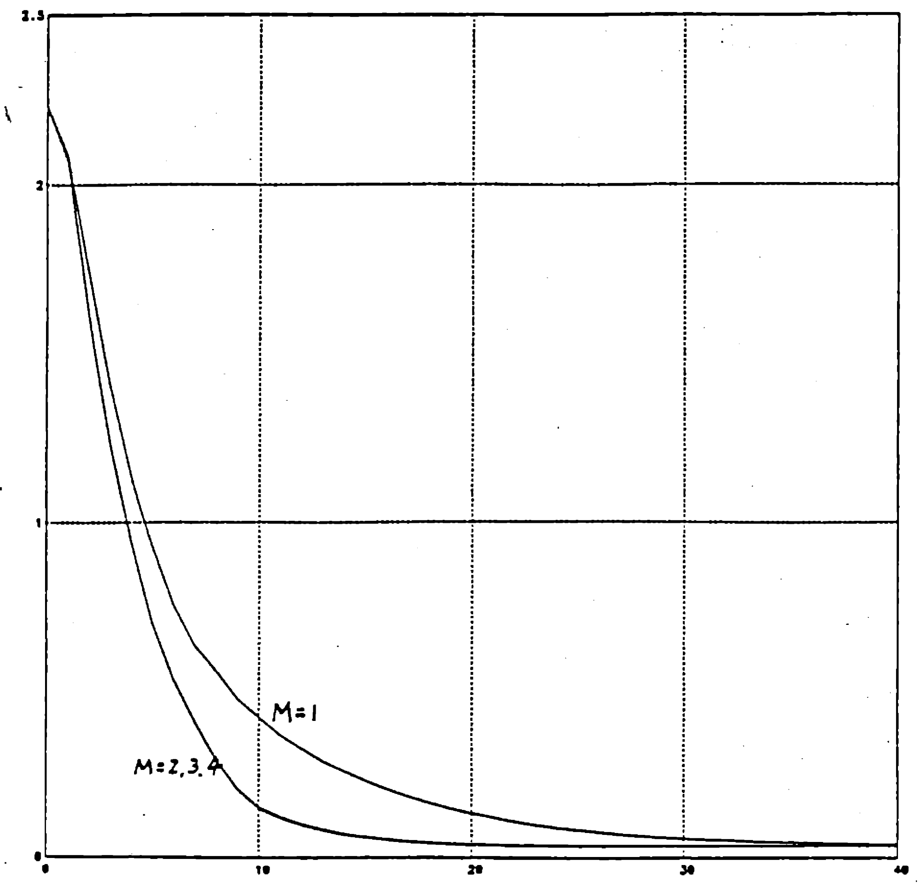
We use the following two test problems:

Example 4.1: The driving function is $e^{5x}\,[2x(x-1)+y(y-1)(25x^2-5x-8)]$ and the true solution is $e^{5x}x(x-1)y(y-1)$. In this case, the solution is a smooth function with a wideband 2-D Fourier spectrum which is concentrated in the region where k_x and k_y are small.

Example 4.2: The driving function is $-74\pi^2\sin(5\pi x)\sin(7\pi y)$ and the true solution is $\sin(5\pi x)\sin(7\pi y)$. This corresponds to the case when the solution is a rapidly oscillatory function containing a single Fourier component at $(k_x, k_y) = (5,7)$.

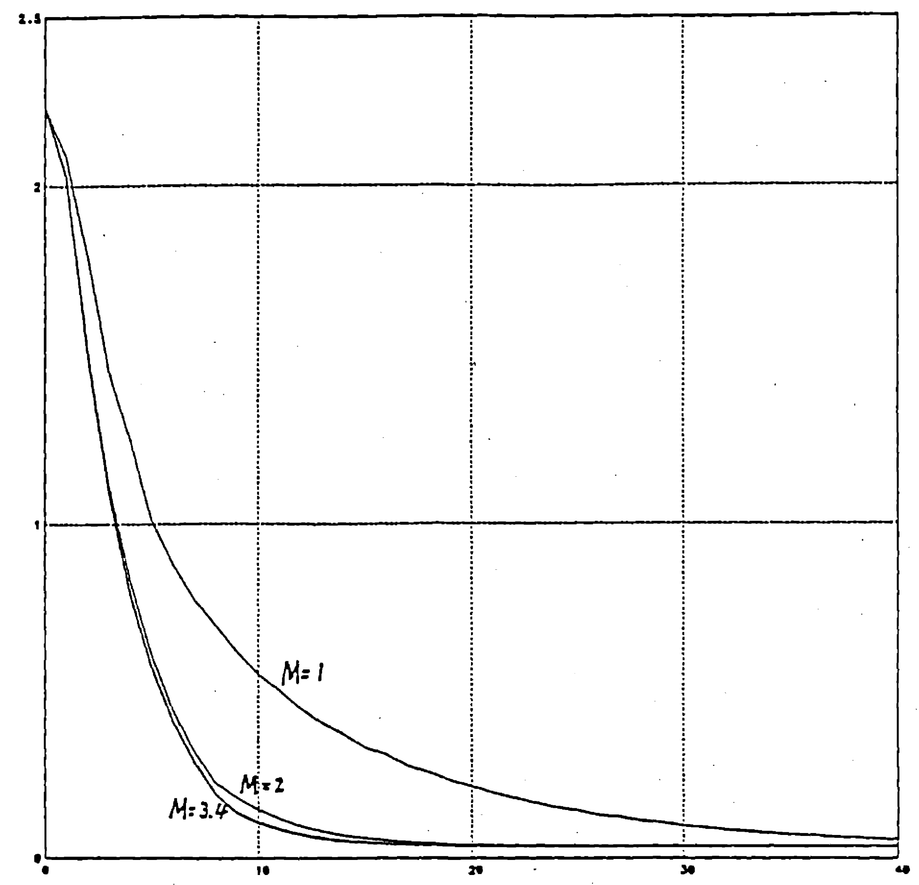
The computed results are shown in Figures 4.4 and 4.5, where we plot the maximum error at each iteration as a function of the number of block SOR iterations. Each curve is parameterized by the number M of point SOR iterations that we have used. It is almost impossible to distinguish the curves with $M = 2,3,4$ for computational order (a) in both examples. Hence, it is reasonable to choose $M = 2$ in this case. When the computational order (b) is applied to the first example, where the solution contains low frequency components, the curve for $M = 3$ is slightly better than for $M = 2$. Nevertheless, the difference is very small. For the second example, the curves with $M = 2,3,4$ are in fact not distinguishable. Thus, for computational order (b), it is still preferable to choose $M = 2$, since less computations are required.

Finally, to demonstrate the convergence rate of the 2-level SOR method, we choose another test problem which has zero driving function, zero boundary conditions, and $x(x-1)y(y-1)$ as the initial value. The same test problem was also considered in [2]. This is in fact a homogeneous Laplacian equation, and its solution is zero. In Figure 4.6, we plot the 2-norm of the error versus the effective number (n_{eff}) of iterations for the above two computational orders and $M=2$. The results show that the 2-level SOR method with computational order (b) is better than that with order (a).

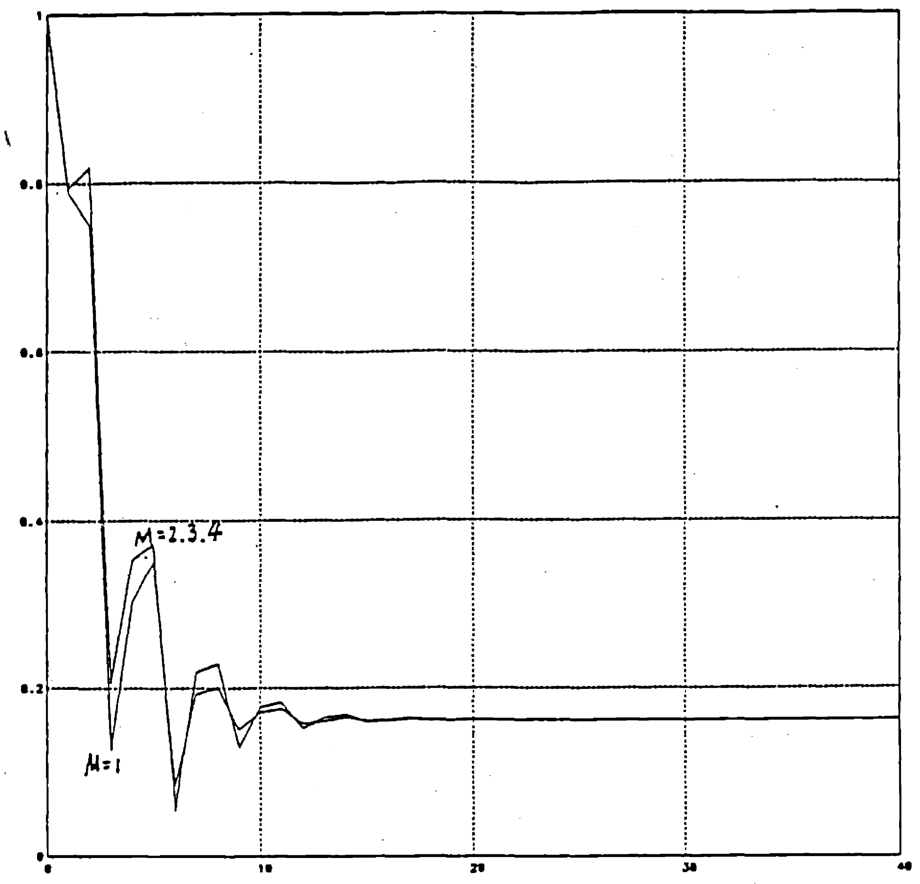


(a)

Figure 4.4: Computer simulation results for Example 4.1 with the computational orders (a) { red → orange → black → green } and (b) { red → black → green → orange }. The x-axis is the number of 1st-level block iterations and the y-axis is the maximum error at each iteration.

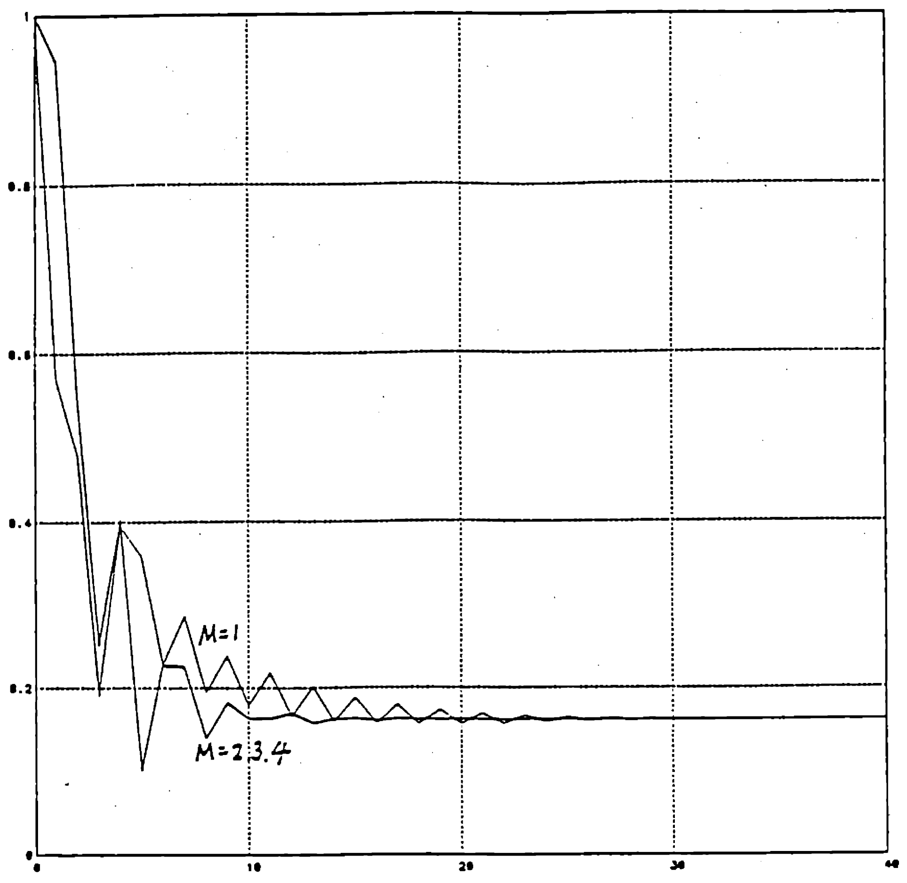


(b)



(a)

Figure 4.5: Computer simulation results for Example 4.2 with the computational orders (a) { red → orange → black → green } and (b) { red → black → green → orange }. The x-axis is the number of 1st-level block iterations and the y-axis is the maximum error at each iteration.



(b)

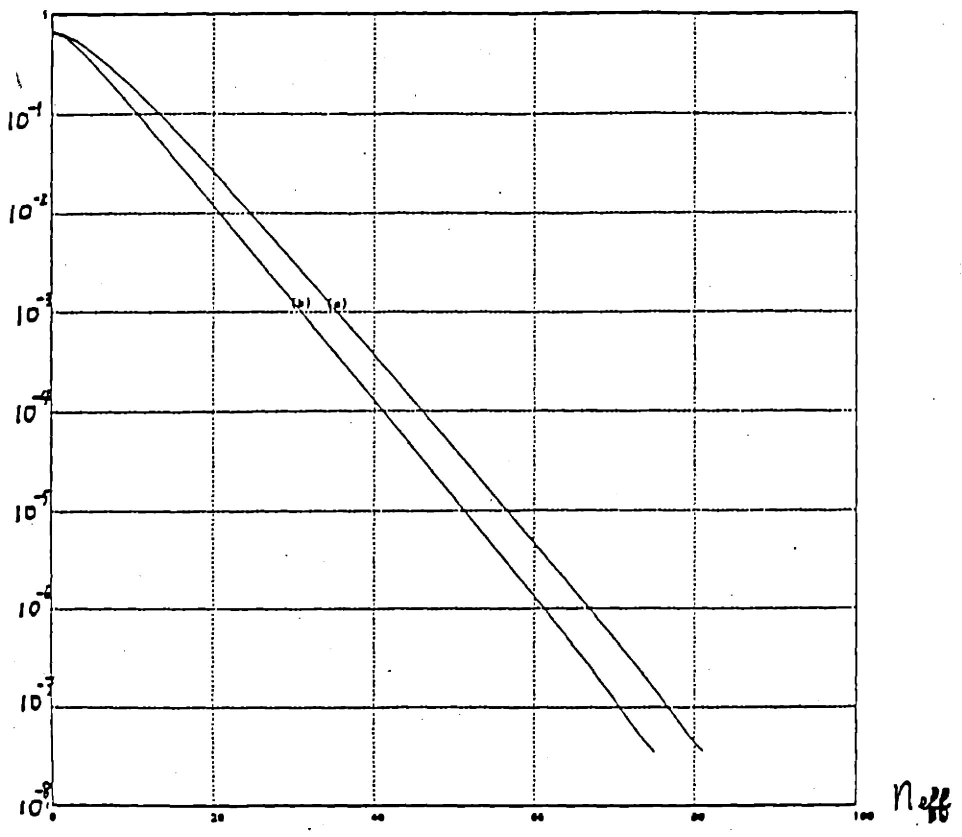


Figure 4.6: Convergence history (2-norm of the error versus the number of effective iterations) for the computational orders (a) { red → orange → black → green } and (b) { red → black → green → orange } with $M=2$. The driving function is zero and the initial value is $x(x-1)y(y-1)$.

4.7 Conclusions and Extensions

The 2-level SOR iteration method presented here can be generalized easily to higher-dimensional problems. A 3-level 8-color SOR scheme is described as follows. Consider a nondegenerate 27-point discretization of the 3-D Poisson equation. Suppose that each grid point is indexed by (n_x, n_y, n_z). We can label these points with 8 colors depending on whether n_x, n_y, and n_z are even or odd. Following a procedure similar to the one used in Section 4.3, we transform the discretized system from the space domain to the frequency domain so that in the frequency domain we obtain an discretization matrix which is block diagonal with 8×8 block matrices along the diagonal. Each of these blocks describes the coupling of the Fourier components of the 8 colors at a fixed frequency. Since the discretization scheme is nondegenerate, each 8×8 matrix block is full. In order to apply the SOR method for each of these 8×8 matrices, we can block partition them into 4×4 submatrices. This results in a 1st level block SOR iteration. However, the 1st level block SOR iteration requires inverting 4×4 full matrices, which can be accomplished by performing several 2nd level block SOR and 3rd level point SOR iterations. Note that both the 2nd level block SOR and 3rd level point SOR iterations require a constant number of steps to converge. The total number of iterations required by the above 3-level SOR method, which is $O(\frac{1}{h})$, is therefore determined primarily by the convergence rate of the 1st level block SOR iteration.

There are many different possible computational orders for the above 3-level SOR procedure. A typical one can be chosen as follows. At the 1st level, we can distinguish two big blocks depending on whether $(n_x + n_y + n_z)$ is even or odd. At the 2nd level, within each big block, points are further divided into two smaller blocks according to whether $(n_x + n_y)$ is even or odd. Finally, at the 3rd level, each color can be separated from each other.

It is straightforward to generalize the above procedure to obtain an l-D l-level 2^l-color SOR method. Here, we have considered the case where $l = 2$.

Another generalization of interest would be to extend the 2-level SOR iteration procedure described in this chapter to PDEs with space-varying coefficients. It is natural in this context to combine the 2-level SOR method discussed here with the local relaxation procedure developed in [4], [6] and [9]. The main idea of the local relaxation method can be roughly stated as follows. Each local finite difference equation is viewed as if it were homogeneous over the entire problem domain so that at each point a local relaxation parameter is determined on the basis of the local coefficients of the PDE and of the boundary conditions for the whole domain. Hence, a 2-level local relaxation method would use the local coefficients and boundary conditions to choose optimal local block and point relaxation parameters at each grid point, so that different grid points would have therefore different block and point relaxation parameters.

Note also that the pure frequency domain approach that we adopted in this chapter depends heavily on the multicolor partitioning scheme. The relation existing between the single-level rowwise and multicolor SOR methods for the 5-point stencil and the 9-point stencil cases can be explained by introducing a tilted grid [10][2]. There does not seem to be an easy way to apply the tilted grid concept to obtain a 2-level rowwise SOR method. Note however that the multicolor and rowwise SOR iterations usually have the same efficiency [1]. In addition, the multicolor SOR method is especially attractive for parallel processing [1][3][9].

In this chapter, we have transformed the system of equations for the discretized PDE from the space domain to the frequency domain so that we were able to interpret the SOR method from a new point of view. This new formulation has helped us to design a 2-level SOR method with optimal block and point relaxation parameters. The resulting 2-level 4-color SOR method for the 9-point stencil discretization of the Poisson equation was shown to be efficient with spectral radius $1 - Ch$, and numerical examples confirm our analysis.

4.8 References

[1] L. Adams and H. F. Jordan, "Is SOR Color-Blind," *SIAM J. Sci. Stat.*, vol. 7, no. 2, Apr. 1986.

[2] L. Adams, R. J. LeVeque, and D. M. Young, "Analysis of the SOR Iteration for the 9-Point Laplacian," To appear in *SIAM J. Num. Analy.*.

[3] L. Adams and J. M. Ortega, "A Multi-Color SOR Method for Parallel Computation," ICASE report, 82-9, Apr. 1982.

[4] E. F. Botta and A. E. P. Veldman, "On Local Relaxation Methods and Their Application to Convection-Diffusion Equations," *Journal of Computational Physics*, vol. 48, pp. 127-149, 1981.

[5] G. Dahlquist, A. Bjorck, and N. Anderson, *Numerical Methods*. Englewood Cliffs, N.J. : Prentice-Hall, Inc. , 1974.

[6] L. W. Erhlich, "The Ad-hoc SOR Method: a Local Relaxation Scheme," in *Elliptic Problem Solvers II*, Academic Press, 1984, pp. 257-269.

[7] S. P. Frankel, "Convergence Rates of Iterative Treatments of Partial Differential Equations," *Math. Tables Aids Comput.*, vol. 4, pp. 65-75, 1950.

[8] P. R. Garabedian, "Estimation of the Relaxation Factor for Small Mesh Size," *Math. Tables Aids Comput.*, vol. 10, pp. 183-185, 1956.

[9] C.-C. J. Kuo, B. C. Levy, and B. R. Musicus, "A Local Relaxation Method for Solving Elliptic PDEs on Mesh-Connected Arrays," *SIAM J. Sci. Stat. Comp.*, to appear.

[10] R. J. LeVeque and L. N. Trefethen, "Fourier Analysis of the SOR Iteration," Numerical Analysis Report 86-6, Department of Mathematics, MIT, Cambridge, MA., also ICASE Report 86-93, Sep. 1986.

[11] R. S. Varga, *Matrix Iterative Analysis*. Englewood Cliffs, N.J. : Prentice-Hall, Inc. , 1962.

[12] D. M. Young , "Iterative Methods for Solving Partial Differential Equations of Elliptic Type ," Doctoral Thesis , Harvard University , 1950.

[13] D. M. Young, *Iterative Solution of Large Linear Systems*. New York, N.Y.: Academic Press, Inc., 1971.

PART III : MULTIGRID METHODS

The third part of this thesis focuses on multigrid methods for the solution of elliptic PDEs.

A two-color Fourier analytical approach is proposed in Chapter 5 to analyze and design multigrid methods which rely on the red/black Gauss-Seidel smoothing iteration. In this approach, Fourier components in the high wavenumber region are folded into the low wavenumber region so that the coupling between the low and high Fourier components is transformed into a coupling between components of red and black computational waves in the low wavenumber region. With this new tool, we develop a two-color multigrid direct solver for the 1D Poisson equation. For the 2D case, we show that the two-color two-grid method asymptotically reduces to a one-color two-grid method whose physical mechanism is more transparent than for its original two-color form. Several design issues such as rearranging the smoothing order and smoothing with a relaxation parameter $\omega \neq 1$ are also studied from the same viewpoint.

Chapter 5 : Two-color Multigrid Methods

5.1 Introduction

It is well known that the multigrid method which employs the red/black Gauss-Seidel smoothing iteration provides a very effective way of solving elliptic PDEs [3][8]. The red/black relaxation scheme is also attractive for parallel computation [2]. However, the mechanism of this method is not as transparent as for methods which use other types of smoothers such as the damped Jacobi iteration [6][8]. Through the red/black Gauss-Seidel iteration, low and high wavenumber components of the solution are coupled together, so that some high wavenumber components are in fact primarily computed by the coarse-grid correction procedure. Therefore, it is more difficult to give a physical explanation of this phenomenon. A two-grid analysis of this method for a model problem consisting of the Poisson equation on the unit square with Dirichlet boundary conditions has been performed by Stüben and Trottenberg [8]. Their analysis is mathematically so involved that it provides little insight into the mechanism of this method. The objective of this chapter is to use a variant of Fourier analysis called the *two-color* Fourier analysis to clarify the physical mechanism of the red/black Gauss-Seidel multigrid method and to investigate several design issues such as rearranging the smoothing order, and smoothing with a relaxation parameter $\omega \neq 1$.

The red/black relaxation operator and the restriction and interpolation operators are linear *periodic* operators. A straightforward Fourier analysis does not apply since they are spatially dependent. Nevertheless, the *periodic* property can be exploited to reformulate the conventional Fourier analysis as a two-color Fourier analysis as presented in Chapters 3 and 4. From this new viewpoint, components in the high wavenumber region are folded into the low wavenumber region so that there exist two, i.e. red and black, computational waves in the low wavenumber region. The coupling between the low and high conventional Fourier components is therefore transformed into a coupling between red and black computational waves with the same wavenumber in the low wavenumber region. With this new Fourier tool, the spectral representation of every operator in the two-grid analysis can be easily derived and interpreted. Then, we show that the two-color two-grid method asymptotically reduces to a one-color two-grid method which is easier to analyze than in its original two-color form. Although our analysis is different from that of Stüben and Trottenberg [8], it turns out, without surprise, that they are mathematically equivalent and lead to the same results.

The two-color Fourier analysis not only serves as an analytical tool but is also a useful design tool. This is particularly evident for the 1D problem, for which the two-color two-grid Fourier analysis is used to design a fast direct method. For 2D problems, we show how to rearrange the smoothing order in such a way that some computational work required in the standard restriction and interpolation procedures can be saved without impairing the convergence rate of the overall algorithm.

Despite the fact that the red/black SOR iteration is an effective single grid solution method, its usefulness in the context of multigrid methods remains doubtful [6][8]. From numerical experiments, we find that the red/black SOR iteration with a relaxation parameter ω slightly greater than 1, can indeed improve the convergence rate. Practically speaking, the introduction of the relaxation parameter ω increases the required computational work, and the improvement of the convergence rate may not offset the additional computational cost. The practical use of such a multigrid method is therefore questionable. Nevertheless, from a theoretical point of view, this is an interesting problem since it provides a link between single-grid and multigrid solution methods. We propose a simplified analysis to explain this phenomenon and to predict the optimal relaxation parameter.

This chapter is organized as follows. The 1D problem is studied in Section 5.2. The analysis and design of 2D two-grid methods is presented in Sections 5.3 and 5.4 respectively. Concluding remarks and extensions are given in Section 5.5.

5.2 Analysis and design of 1D two-color multigrid method

Consider a $(h, 2h)$ two-grid method for solving the discretized 1-D Poisson equation on $\Omega=[0,1]$ with boundary values $u(0)$ and $u(1)$, i.e.

$$\frac{1}{h^2}(u_{n-1} - 2u_n + u_{n+1}) = f_n \;, \quad n = 1, 2, \cdots, N-1 \,, \tag{1}$$

where u_n is the estimate of $u(nh)$, h is the grid spacing, and $N = \frac{1}{h}$ is even. The difference between the exact solution $\bar{u}_n$ and the estimate u_n is the error $e_n = u_n - \bar{u}_n$. For a two-grid method, the error equation can be written as

$$\mathrm{e}^{new} = M_h^{2h}\mathrm{e}^{old} \,,$$

where $\mathrm{e} = (e_1, \cdots, e_{N-1})^T$ and M_h^{2h} is the two-grid iteration operator [8],

$$M_h^{2h} = S_h^{\nu_2} K_h^{2h} S_h^{\nu_1} \,,$$

where S_h is the smoothing operator (smoother) for the h-grid Ω_h, ν_1 and ν_2 are the numbers of presmoothing and postsmoothing iterations, and K_h^{2h} is the coarse-grid correction operator (coarse-grid corrector)

$$K_h^{2h} = I_h - I_{2h}^{h} L_{2h}^{-1} I_h^{2h} L_h \,,$$

and where I_h, I_{2h}^h, L_{2h}, I_h^{2h}, L_h are the identity, interpolation, coarse-grid Laplacian, restriction, and fine-grid Laplacian operators respectively.

5.2.1 Two-color Fourier analysis

The analysis of M_h^{2h} is often performed in the wavenumber domain so that we consider the coefficients $\hat{e}_k$, $1 \leqslant k \leqslant N-1$, of the Fourier expansion

$$e_n = \sum_{k=1}^{N-1} \hat{e}_k \sin(k\pi nh) \,. \tag{2}$$

The decomposition (2) is particularly convenient for understanding multigrid methods which employ the damped Jacobi smoothing iteration [8]. However, when we use the red/black Gauss-Seidel smoothing iteration, the Fourier components $\hat{e}_k$ and $\hat{e}_{N-k}$ are coupled together. In this paper, a modified Fourier analysis is introduced to analyze this type of smoother. As usual, we call grid points with even and odd indices the red and black points. Errors at red and black points form red and black sequences, which can be expanded in Fourier series as

$$e_n = \sum_{k=1}^{\frac{N}{2}-1} \hat{r}_k \sin(k\,\pi n h) \quad , n \quad \text{even} , \tag{3a}$$

$$e_n = \sum_{k=1}^{\frac{N}{2}} \hat{b}_k \sin(k\,\pi n h) \quad , n \quad \text{odd} . \tag{3b}$$

It is straightforward to see that the Fourier components of the red and black sequences are related to the Fourier components of the complete sequence e_n via

$$\begin{pmatrix} \hat{r}_k \\ \hat{b}_k \end{pmatrix} = \frac{1}{2} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{pmatrix} \hat{e}_k \\ \hat{e}_{N-k} \end{pmatrix} , \quad 1 \leqslant k \leqslant \frac{N}{2}-1 \quad \text{and} \quad \hat{b}_{\frac{N}{2}} = \hat{e}_{\frac{N}{2}} .$$

The decomposition (3), called the two-color Fourier analysis, is particularly convenient for operators operating on grid points on a periodical basis.

For example, consider a Jacobi relaxation operation operating at the red points only,

$$e_n^{new} = \frac{1}{2}[\, e_{n-1}^{old} + e_{n+1}^{old} \,] \quad n \quad \text{even} , \quad e_n^{new} = e_n^{old} \quad n \quad \text{odd} .$$

In the spectral domain, the matrix representation of this iteration describing its action on $(\, \hat{r}_k \,, \hat{b}_k \,)^T$, $1 \leqslant k \leqslant \frac{N}{2}-1$, is given by

$$\hat{S}_{h,r}(\theta) = \begin{bmatrix} 0 & \cos\theta \\ 0 & 1 \end{bmatrix} , \qquad \theta = k\,\pi h \;.$$

For $k = \frac{N}{2}$, $\hat{S}_{h,r}(\frac{\pi}{2})$ is a mapping from $\hat{b}_{\frac{N}{2}}$ onto itself and equals to 1. Similarly, a

Jacobi relaxation operation operating at the black points only gives

$$\hat{S}_{h,b}(\theta) = \begin{bmatrix} 1 & 0 \\ \cos\theta & 0 \end{bmatrix}, \quad 0 < \theta < \frac{\pi}{2}, \quad \text{and} \quad \hat{S}_{h,b}(\frac{\pi}{2}) = 0 . \tag{4}$$

Hence, the spectral representation of the red/black Gauss-Seidel iteration can be easily obtained as

$$\hat{S}_{h,r/b}(\theta) = \hat{S}_{h,b}(\theta)\,\hat{S}_{h,r}(\theta) = \begin{bmatrix} 0 & \cos\theta \\ 0 & \cos^2\theta \end{bmatrix} \quad 0 < \theta < \frac{\pi}{2}, \quad \text{and} \quad \hat{S}_{h,r/b}(\frac{\pi}{2}) = 0 .$$

5.2.2 A two-color multigrid direct solver

Now, let us study the coarse-grid corrector K_h^{2h} by using the basis $(\hat{r}_k, \hat{b}_k)^T$. Let the restriction operator I_h^{2h} and the interpolation operator I_{2h}^h be

$$I_h^{2h} : \quad | \frac{1}{4}, \frac{1}{2}, \frac{1}{4} |_h^{2h} \quad \text{and} \quad I_{2h}^h : \quad | c, 1, c |_{2h}^h , \tag{5}$$

where c is an arbitrary constant. Since points of the coarse grid coincide exactly with red points of the fine grid for the 1D case, $\hat{I}_h^{2h}(\theta)$, which is a mapping from $(\hat{r}_k, \hat{b}_k)^T$ to $\hat{r}_k$, and $\hat{I}_{2h}^h(\theta)$, a mapping from $\hat{r}_k$ to $(\hat{r}_k, \hat{b}_k)^T$, assume the following simple forms,

$$\hat{I}_h^{2h}(\theta) = [\,\frac{1}{2}, \frac{\cos\theta}{2}\,] \quad \text{and} \quad \hat{I}_{2h}^h(\theta) = \begin{bmatrix} 1 \\ 2c\cos\theta \end{bmatrix} .$$

In the red/black spectral domain, the $2h$-grid, the h-grid discretized Laplacian operators and the identity matrix are represented respectively by

$$\hat{L}_{2h}(\theta) = \frac{2(\cos 2\theta - 1)}{(2h)^2}, \quad \hat{L}_h(\theta) = \frac{2}{h^2} \begin{bmatrix} -1 & \cos\theta \\ \cos\theta & -1 \end{bmatrix}, \quad \hat{I}_h(\theta) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} .$$

Hence, we obtain

$$\hat{K}_h^{2h}(\theta) = \hat{I}_h(\theta) - \hat{I}_{2h}^h(\theta)\hat{L}_{2h}^{-1}(\theta)\hat{I}_h^{2h}(\theta)\hat{L}_h(\theta) = \begin{bmatrix} 0 & 0 \\ -2c\cos\theta & 1 \end{bmatrix} . \tag{6}$$

Equation (6) shows that all red computational waves are eliminated by the coarse-grid corrector K_h^{2h}. Suppose that we are able to eliminate the effect of all black computational waves by some smoothing operation. Then, one coarse-grid correction followed by such a smoothing operation is sufficient for solving the two-grid problem exactly. From (4), we know that a simple Jacobi iteration at the black points, i.e. $S_{h,b}$, serves this purpose. Consequently, by choosing

$$M_h^{2h} = S_{h,b}\ K_h^{2h}\ , \quad \text{with} \quad K_h^{2h} = I_h - I_{2h}^{h} L_{2h}^{-1} I_h^{2h} L_h\ , \tag{7}$$

$\hat{M}_h^{2h}(\theta)$ are 2×2 zero matrices for all $0<\theta<\frac{\pi}{2}$. Besides, $\hat{M}_h^{2h}(\frac{\pi}{2})$ is also zero. Thus, the two-grid method (7) is exact.

5.2.3 Modification and generalization

Although the above analysis is independent of the value c, the choice $c = 0$ saves computational work and, therefore, is preferable in practice. It is possible to reduce the computational work of (7) further by using $S_{h,r}$ or $S_{h,b}$ as presmoother. Depending on whether we use $S_{h,r}$ or $S_{h,b}$, the residues at the red or black points are zero, and in this case the restriction operator I_h^{2h} in (5) can be replaced either by

$$[\ \frac{1}{4}\ ,0\ ,\ \frac{1}{4}\ ]_h^{2h} \qquad \text{or} \qquad [\ 0\ ,\ \frac{1}{2}\ ,\ 0\ ]_h^{2h}\ .$$

In particular, if we use $S_{h,b}$ as presmoother and let $c = 0$, a modified two-color two-grid direct solver can be described as follows:

(1) Perform a Jacobi iteration at black points.
(2) Calculate residues at the red points and multiply them by $\frac{1}{2}$.
(3) Solve the system of residue equations on the $2h$ -grid and add the coarse-grid solution back to the original values at the red points.
(4) Perform a Jacobi iteration at black points.

This algorithm corresponds to the following two-grid operator

$$M_h^{2h} = S_{h,b} \, (I_{h,r} - L_{2h}^{-1} \frac{1}{2} L_{h,r}) \, S_{h,b} \ , \tag{8}$$

where $L_{h,r}$ is the restriction of the discretized Laplacian operator to the red points of the h-grid, and $I_{h,r}$ is the identity operator for the red points. For $0 < \theta < \frac{\pi}{2}$, the spectral representation of $L_{h,r}$ and $I_{h,r}$ is given by

$$\hat{L}_{h,r}(\theta) = \frac{2}{h^2} \begin{bmatrix} -1 & \cos\theta \\ 0 & 0 \end{bmatrix} , \qquad \hat{I}_{h,r}(\theta) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} .$$

Note that the calculation of the residue takes the same amount of work as the smoothing operation at every grid point. We compute the residues at all grid points in (7) while we perform the smoothing operation at one half of the grid points and compute the residue at the other half of the grid points in (8). The saving comes from the fact that a 3-point averaging operation is needed by (7) and that only a multiplication of $\frac{1}{2}$ is required by (8). The saving in the restriction and interpolation procedures will be generalized to the 2D case in Section 5.4.1.

A two-color L-grid direct solver ($L > 2$) can be defined by using the above two-color two-grid method recursively, i.e.

$$M_h^{2h} = S_{h,b} \, (I_{h,r} - X_{2h} \frac{1}{2} L_{h,r}) \, S_{h,b} \ , \tag{9a}$$

with

$$X_h = M_h^{2h} \, , h = \frac{1}{2^l} \, , 2 \leqslant l \leqslant L-1 \ , \ \text{and} \ \ X_h = L_h^{-1} \, , h = \frac{1}{2} \, . \tag{9b}$$

It can be proved by induction that (9) is a direct method for the system of equations (1).

There exists no analog of equation (6) for 2D problems so that there is no 2D direct solver corresponding to the one described above. However, a relation similar to

(6) holds in the low wavenumber region which means that the 2D coarse-grid corrector can reduce errors of low wavenumber components effectively.

5.3 Analysis of 2D two-color multigrid method

In this section, the two-color Fourier analysis is used to analyze a $(h, 2h)$ two-grid method. We choose the discretized 2D Poisson equation,

$$\frac{1}{h^2}\left(u_{n_x-1,n_y} + u_{n_x+1,n_y} + u_{n_x,n_y-1} + u_{n_x,n_y+1} - 4\,u_{n_x,n_y} \right) = f_{n_x,n_y} \,, \quad 1 \leqslant n_x, n_y \leqslant N-1 \,, \qquad (1)$$

where u_{n_x,n_y} is given for n_x, $n_y = 0$ or N, and $N = \frac{1}{h}$ is even, as an example. Since this two-grid algorithm has been analyzed by Stüben and Trottenberg with the standard Fourier analysis [8], the physical interpretation associated to the two-color Fourier analysis, rather than the specific mathematical result that we derive, will be emphasized in the following discussion.

5.3.1 2D two-color Fourier analysis

The errors e_{n_x,n_y} associated to (1) can be expanded as

$$e_{n_x,n_y} = \sum_{k_x=1}^{N-1}\sum_{k_y=1}^{N-1} \hat{e}_{k_x,k_y} \sin(k_x \pi n_x h) \sin(k_y \pi n_y h) \,.$$

We divide grid points with indices $\mathbf{n} = (n_x, n_y)$ into red and black points, depending on whether $n_x + n_y$ is even or odd. Then, errors at red and black points define red and black sequences, which can be expanded as

$$e_{n_x,n_y} = \sum_{\mathbf{k} \in K_r} \hat{r}_{k_x,k_y} \sin(k_x \pi n_x h) \sin(k_y \pi n_y h) \,, \qquad n_x + n_y \quad \text{even} \,,$$

$$e_{n_x,n_y} = \sum_{\mathbf{k} \in K_b} \hat{b}_{k_x,k_y} \sin(k_x \pi n_x h) \sin(k_y \pi n_y h) \,, \qquad n_x + n_y \quad \text{odd} \,,$$

where $K_r = K$ and $K_b = K \cup \{ (\frac{N}{2}, \frac{N}{2}) \}$, and where

$$K = \{ (k_x, k_y) \in I^2 : \; k_x + k_y \leqslant N-1 \,,\, k_x, k_y \geqslant 1 \quad \text{or} \quad k_y = N - k_x \,,\; 1 \leqslant k_x \leqslant \tfrac{N}{2} - 1 \} \,,$$

For $\mathbf{k} \in K$, we denote $(N-k_x, N-k_y)$ by $\mathbf{k}'$. It is straightforward to check that $\hat{r}_{\mathbf{k}}$,

$\hat{b}_{\mathbf{k}}, \hat{e}_{\mathbf{k}}, \hat{e}_{\mathbf{k}'}$ are related via

$$\begin{bmatrix} \hat{r}_{\mathbf{k}} \\ \hat{b}_{\mathbf{k}} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} \hat{e}_{\mathbf{k}} \\ \hat{e}_{\mathbf{k}'} \end{bmatrix}, \quad \mathbf{k} \in K \quad \text{and} \quad \hat{b}_{\mathbf{k}} = \hat{e}_{\mathbf{k}}, \quad \mathbf{k} = (\tfrac{N}{2}, \tfrac{N}{2}).$$

The original and the folded two-color Fourier domains are depicted in Figure 5.1. Note that K_r and K_b differs only by a single element $(\frac{N}{2},\frac{N}{2})$ and, therefore, at the wavenumber $(\frac{N}{2},\frac{N}{2})$ we have only a scalar $\hat{b}_{\frac{N}{2},\frac{N}{2}}$. As before, we define $\theta = (\theta_x, \theta_y) = (k_x \pi h, k_y \pi h)$ and Θ denotes the set of θ whose corresponding $\mathbf{k}$ belongs to K.

5.3.2 Analysis of two-color two-grid method

We consider the two-grid iteration matrix with one red/black Gauss-Seidel iteration

$$M_h^{2h} = K_h^{2h} S_{h,b} S_{h,r} \;\; (\text{ or } S_{h,b} S_{h,r} K_h^{2h}\,), \qquad K_h^{2h} = I_h - I_{2h}^{h} L_{2h}^{-1} I_h^{2h} L_h \,, \tag{2}$$

where L_h and L_{2h} are the 5-point discretizations of the Laplacian on the h and $2h$ grids, and I_h^{2h} and I_{2h}^{h} are the full-weighting restriction and linear interpolation operators, given by

$$I_h^{2h} : \left| \begin{matrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{matrix} \right|_h^{2h} . \tag{3a}$$

and

$$I_{2h}^{h} : \left| \begin{matrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{matrix} \right|_{2h}^{h} . \tag{3b}$$

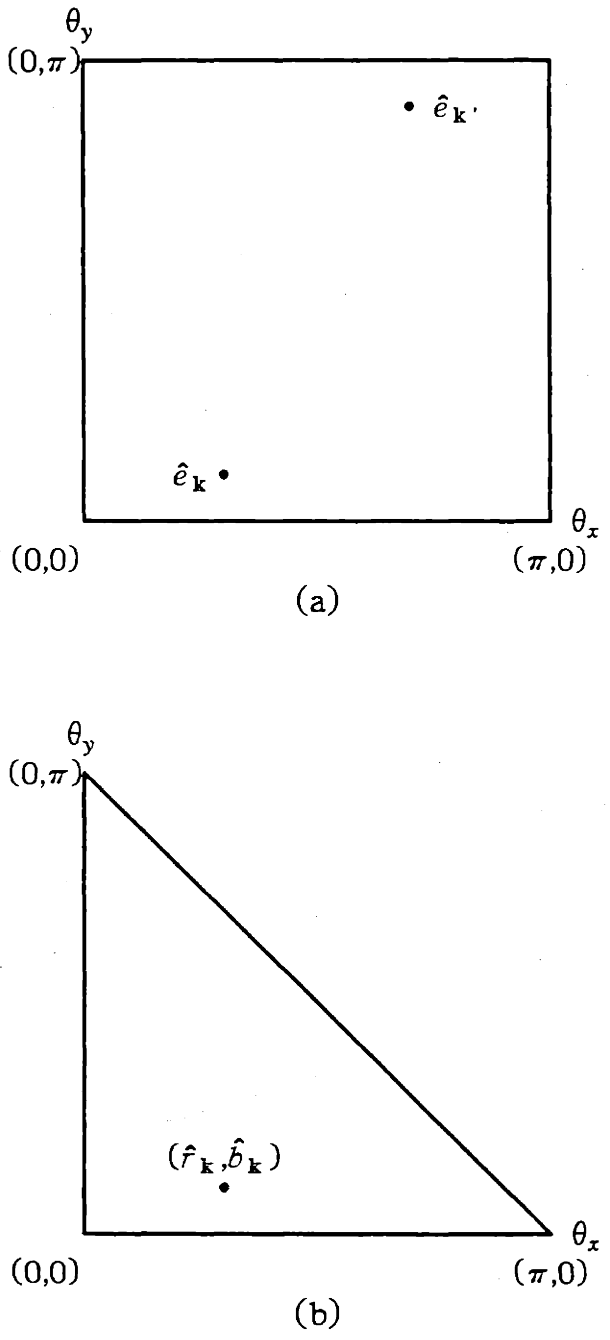


Figure 5.1: (a) Conventional and (b) folded two-color Fourier domains.

The problem is to determine the spectral radius $\rho(M_h^{2h})$ of the two-grid iteration matrix.

Each of the 4×4 wavenumber domain matrices appearing below corresponds to a mapping from a vector space formed by the vector

$$(r_{\mathbf{k}} \, , -r_{\tilde{\mathbf{k}}} \, , b_{\mathbf{k}} \, , -b_{\tilde{\mathbf{k}}} \,)^T \, ,$$

onto itself, where

$$\mathbf{k} = (k_x, k_y) \quad 1 \leqslant k_x, k_y < \frac{N}{2}, \quad \tilde{\mathbf{k}} = \begin{cases} (N-k_x, k_y) & \text{if } k_x \geqslant k_y \\ (k_x, N-k_y) & \text{if } k_x < k_y \end{cases} .$$

We also use the abbreviations

$$\alpha = \frac{\cos\theta_x + \cos\theta_y}{2} \, , \quad \tilde{\alpha} = \frac{\cos\tilde{\theta}_x + \cos\tilde{\theta}_y}{2} \, , \quad \beta = \cos\theta_x \cos\theta_y \, , \quad \tilde{\beta} = \cos\tilde{\theta}_x \cos\tilde{\theta}_y \, .$$

(1) **Smoothing**

For $\theta_x \, , \theta_y < \frac{\pi}{2}$, the wavenumber domain matrix corresponding to the red/black Gauss-Seidel iteration is

$$\hat{S}_{h,r/b}(\theta) = \hat{S}_{h,b}(\theta) \, \hat{S}_{h,r}(\theta) = \begin{bmatrix} I & 0 \\ J & 0 \end{bmatrix} \begin{bmatrix} 0 & J \\ 0 & I \end{bmatrix} = \begin{bmatrix} 0 & J \\ 0 & J^2 \end{bmatrix} , \tag{4}$$

where 0 is the 2×2 zero matrix, I is the 2×2 identity matrix, and

$$J = \begin{bmatrix} \alpha & 0 \\ 0 & \tilde{\alpha} \end{bmatrix} .$$

When θ_x or θ_y is equal to $\frac{\pi}{2}$, $\hat{S}_{h,r/b}$ is a 2×2 matrix

$$\hat{S}_{h,r/b}(\theta) = \begin{bmatrix} 0 & \alpha \\ 0 & \alpha^2 \end{bmatrix} ,$$

which is a mapping from $(\hat{r}_{\mathbf{k}}, \hat{b}_{\mathbf{k}})^T$ to $(\hat{r}_{\mathbf{k}}, \hat{b}_{\mathbf{k}})^T$. Finally, for $\theta_x = \theta_y = \frac{\pi}{2}$, $\hat{S}_{h,r/b} = 0$, which is a mapping from $\hat{b}_{\mathbf{k}}$ to itself.

Note that when the first partial step of the red/black Gauss-Seidel iteration, i.e. the Jacobi iteration at red points, is performed, the original values of the red points are discarded and, hence, the computational process that follows is only determined by the initial values of the black points. This observation is the basis for reducing the two-color analysis to a one-color analysis.

(2) **Coarse-grid correction**

Let us first consider the case θ_x , θ_y $< \frac{\pi}{2}$. The wavenumber domain matrices for operators I_h , L_h and L_{2h}^{-1} in (2) can be written as

$$\hat{I}_h(\theta) = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} , \quad \hat{L}_h(\theta) = \frac{4}{h^2} \begin{bmatrix} -I & J \\ J & -I \end{bmatrix} , \quad \hat{L}_{2h}^{-1}(\theta) = \frac{h^2}{2\delta} , \delta = 2\alpha^2 - \beta - 1 . \qquad (5)$$

In (4) and (5), there is no coupling between vectors $(\hat{r}_{\mathbf{k}}, \hat{b}_{\mathbf{k}})^T$ and $(\hat{r}_{\tilde{\mathbf{k}}}, \hat{b}_{\tilde{\mathbf{k}}})^T$. The coupling between them comes from the full-weighting restriction and linear interpolation operations, which are more complicated than in the 1D case since the coarse-grid points do not coincide any longer with the red points of the h -grid.

The decomposition, shown in Figure 5.2 and commonly used in the multirate signal processing context [4], is very useful for understanding the physical mechanism of interpolation and restriction operators, and for deriving their wavenumber domain matrices. Conceptually, we decompose the restriction procedure into two steps.

Step 1: lowpass filtering (or averaging) at every point of Ω_h , where the weighting coefficients are specified by stencil (3a).

Step 2: down-sampling (or injecting) values from Ω_h to Ω_{2h} .

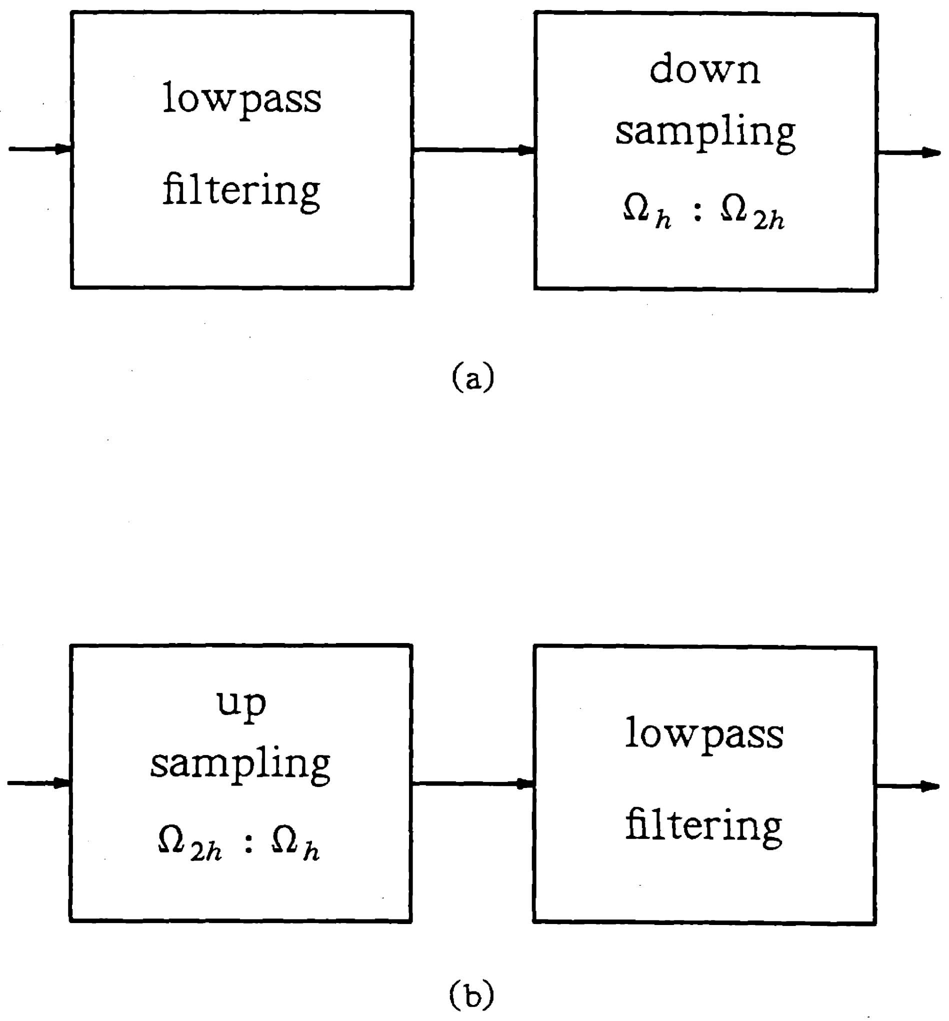

lowpass filtering
down sampling $\Omega_h : \Omega_{2h}$
(a)
up sampling $\Omega_{2h} : \Omega_h$
lowpass filtering
(b)

Figure 5.2: Decomposition of the (a) restriction and (b) interpolation operators.

The interpolation operator I_{2h}^{h} is also decomposed into two steps.

Step 1: up-sampling values from Ω_{2h} to Ω_h, by which we assign 0 to points which belong to $\Omega_h - \Omega_{2h}$

Step 2: lowpass filtering at every point of Ω_h, where the weighting coefficients are specified by stencil (3b).

It is relatively easy to find a wavenumber domain matrix representation for each of the above steps. Combining them together, we obtain

$$\hat{I}_h^{2h}(\boldsymbol{\theta}) = [1\ \ 1\ \ 0\ \ 0] \times \frac{1}{4} \begin{bmatrix} 1+\beta & 0 & 2\alpha & 0 \\ 0 & 1+\tilde{\beta} & 0 & 2\tilde{\alpha} \\ 2\alpha & 0 & 1+\beta & 0 \\ 0 & 2\tilde{\alpha} & 0 & 1+\tilde{\beta} \end{bmatrix} = \frac{1}{4}[1+\beta\ \ 1+\tilde{\beta}\ \ 2\alpha\ \ 2\tilde{\alpha}]\,, \tag{6a}$$

and

$$\hat{I}_{2h}^{h}(\boldsymbol{\theta}) = \begin{bmatrix} 1+\beta & 0 & 2\alpha & 0 \\ 0 & 1+\tilde{\beta} & 0 & 2\tilde{\alpha} \\ 2\alpha & 0 & 1+\beta & 0 \\ 0 & 2\tilde{\alpha} & 0 & 1+\tilde{\beta} \end{bmatrix} \times \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1+\beta \\ 1+\tilde{\beta} \\ 2\alpha \\ 2\tilde{\alpha} \end{bmatrix} . \tag{6b}$$

Thus, in the wavenumber domain, the down-sampling operation adds the high wavenumber component $-\hat{r}_{\tilde{\mathbf{k}}}$ to the low wavenumber component $\hat{r}_{\mathbf{k}}$. This phenomenon is known as *aliasing* [4]. On the other hand, the up-sampling operation duplicates the low wavenumber component $\hat{r}_{\mathbf{k}}$ in the high wavenumber region in the form of $-\hat{r}_{\tilde{\mathbf{k}}}$, which is called *imaging* [4]. The lowpass filters cascaded with the down-sampling and the up-sampling operators are basically used to reduce the aliasing and imaging effects. For example, when θ_x and θ_y are close to 0, $\alpha \approx 1$, $\beta \approx 1$, $\tilde{\alpha} \approx 0$, and $\tilde{\beta} \approx -1$. Hence, the aliasing and imaging effects occurring between $(\hat{r}_{\mathbf{k}}, \hat{b}_{\mathbf{k}})^T$ and $(\hat{r}_{\tilde{\mathbf{k}}}, \hat{b}_{\tilde{\mathbf{k}}})^T$ are substantially eliminated by the associated lowpass filters.

The product $\hat{I}_{2h}^{h}(\theta)\hat{I}_h^{2h}(\theta)$ can be expressed as

$$\hat{I}_{2h}^{h}(\theta)\hat{I}_h^{2h}(\theta)=\frac{1}{8}\begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22}\end{bmatrix}, \quad \text{where} \quad F_{11}=\begin{bmatrix} (1+\beta)^2 & (1+\beta)(1+\tilde{\beta}) \\ (1+\beta)(1+\tilde{\beta}) & (1+\tilde{\beta})^2 \end{bmatrix}, \tag{7}$$

$$F_{12}=F_{21}^T=\begin{bmatrix} 2\alpha(1+\beta) & 2\tilde{\alpha}(1+\beta) \\ 2\alpha(1+\tilde{\beta}) & 2\tilde{\alpha}(1+\tilde{\beta}) \end{bmatrix}, \qquad F_{22}=\begin{bmatrix} 4\alpha^2 & 4\alpha\tilde{\alpha} \\ 4\alpha\tilde{\alpha} & 4\tilde{\alpha}^2 \end{bmatrix}.$$

Therefore, from (5) and (7), we obtain the coarse-grid corrector,

$$\hat{K}_h^{2h}(\theta)=\begin{bmatrix} K_{11} & K_{12} \\ K_{21} & K_{22}\end{bmatrix},$$

where

$$K_{11}=I-\frac{1}{4\delta}(F_{12}J-F_{11}), \qquad K_{22}=I-\frac{1}{4\delta}(F_{21}J-F_{22}),$$

$$K_{12}=\frac{-1}{4\delta}(F_{11}J-F_{12}), \qquad K_{21}=\frac{-1}{4\delta}(F_{22}J-F_{21}),$$

which holds for $\theta_x, \theta_y < \frac{\pi}{2}$. For the remaining cases, we can show that $\hat{K}_h^{2h}(\theta)$ is either the 2×2 identity matrix or 1, depending on whether only one of θ_x, θ_y is $\frac{\pi}{2}$ or both θ_x and θ_y are $\frac{\pi}{2}$.

(3) Two-grid iteration

Combining results in the previous discussion, we find that in the wavenumber domain $M_h^{2h}=K_h^{2h}S_{h,r/b}$ is represented as

$$\hat{M}_h^{2h}(\theta)=\begin{bmatrix} 0 & K_{11}J+K_{12}J^2 \\ 0 & K_{21}J+K_{22}J^2\end{bmatrix}, \qquad \theta_x, \theta_y < \frac{\pi}{2}, \tag{8a}$$

$$\hat{M}_h^{2h}(\theta)=\begin{bmatrix} 0 & \alpha \\ 0 & \alpha^2\end{bmatrix}, \qquad \theta_x \text{ or } \theta_y = \frac{\pi}{2}, \tag{8b}$$

$$\hat{M}_h^{2h}(\theta)=0, \qquad \theta_x=\theta_y=\frac{\pi}{2}. \tag{8c}$$

Therefore, the spectral radius of $\hat{M}_h^{2h}(\theta)$ is

$$\rho(\hat{M}_h^{2h}(\theta)) = \begin{cases} \rho(K_{21}J + K_{22}J^2) & \theta_x\,,\theta_y < \frac{\pi}{2} \\ \alpha^2 & \theta_x \text{ or } \theta_y = \frac{\pi}{2}\,, \\ 0 & \theta_x = \theta_y = \frac{\pi}{2} \end{cases} \tag{9}$$

and finally the spectral radius of the two-grid iteration matrix is

$$\rho(M_h^{2h}) = \max_{\theta_x,\theta_y \leq \frac{\pi}{2}} \rho(\hat{M}_h^{2h}(\theta))\,.$$

The two-to-one color reduction is mathematically clear from equations (8) and (9). Note that the two-grid iteration process $M_h^{2h}(\theta)$ is the combination of two processes

$$M_{12}(\theta) = K_{11}J + K_{12}J^2\,, \qquad M_{22}(\theta) = K_{21}J + K_{22}J^2\,,$$

which describe the evolution from $(b_{\mathbf{k}}, -b_{\bar{\mathbf{k}}})^T$ to $(r_{\mathbf{k}}, -r_{\bar{\mathbf{k}}})^T$ and $(b_{\mathbf{k}}, -b_{\bar{\mathbf{k}}})^T$ respectively. Since the m-fold repetition of M_h^{2h} gives

$$[(\,\hat{M}_h^{2h}(\theta)\,)^m\,]_{12} = M_{12}(\theta)\,M_{22}^{m-1}(\theta)\,, \qquad [(\,\hat{M}_h^{2h}(\theta)\,)^m\,]_{22} = M_{22}^{m}(\theta)\,,$$

the convergence of the two-grid method depends entirely on the process $M_{22}(\theta)$.

The above derivation can be easily generalized to the case with more than one red/black Gauss-Seidel smoothing operation. Suppose that ν_1 and ν_2 such smoothing operations are used respectively for the presmoother and postsmoother, then

$$\rho[M_h^{2h}(\nu_1,\nu_2)] = \rho(S_{h,r/b}^{\nu_2} K_h^{2h} S_{h,r/b}^{\nu_1}) = \rho(K_h^{2h} S_{h,r/b}^{\nu_1+\nu_2})\,,$$

where the last equality comes from the fact $\rho(AB) = \rho(BA)$, and

$$\rho(\hat{M}_h^{2h}(\nu_1,\nu_2,\theta)) = \begin{cases} \rho(K_{21}J^{2\nu-1} + K_{22}J^{2\nu}) & \theta_x\,,\theta_y < \frac{\pi}{2} \\ \alpha^{2\nu} & \theta_x \text{ or } \theta_y = \frac{\pi}{2}\,, \\ 0 & \theta_x = \theta_y = \frac{\pi}{2} \end{cases}$$

where $\nu = \nu_1 + \nu_2$.

Let us examine the matrix

$$M_{eq} = K_{21}J^{2\nu-1}+K_{22}J^{2\nu} , \tag{10}$$

which represents a one-color two-grid iteration process and can be expressed as

$$M_{eq} = JK_{eq}\,J^{2\nu-1} , \tag{11}$$

where

$$K_{eq} = I - \frac{1}{4\delta}J^{-1}F_{21}(J^2-I) = \begin{bmatrix} 1-\frac{(1+\beta)(\alpha^2-1)}{2\delta} & -\frac{(1+\tilde{\beta})(\tilde{\alpha}^2-1)}{2\delta} \\ -\frac{(1+\beta)(\alpha^2-1)}{2\delta} & 1-\frac{(1+\tilde{\beta})(\tilde{\alpha}^2-1)}{2\delta} \end{bmatrix}$$

is the equivalent one-color coarse-grid corrector. Since $\rho(JK_{eq}\,J^{2\nu-1}) = \rho(K_{eq}\,J^{2\nu})$, we see that J^2 can be viewed as the equivalent one-color smoother S_{eq}, which corresponds to two Jacobi relaxation steps for the black component $b_{\mathbf{k}}$.

In [8], Stüben and Trottenberg reduced their analysis to the determination of the largest value among all the spectral radii of matrices $J^{2\nu}K_{eq}$, $0 < \theta_x ,\theta_y < \frac{\pi}{2}$, and a closed form of this quantity has been derived (pp. 104-108). Since the same result holds here, we summarize it as follows

$$\rho[\, M_h^{2h}(\nu=\nu_1+\nu_2)\,] = \begin{cases} \frac{1}{4} & \nu = 1 \\ \frac{1}{2\nu}(\frac{\nu}{\nu+1})^{\nu+1} & \nu \geqslant 2 \end{cases} .$$

In the above expression, the maximum of $\rho[\hat{M}_h^{2h}(\theta)]$ occurs at $\theta = (\frac{\pi}{2},0)$ or $(0,\frac{\pi}{2})$ when $\nu = 1$ and at ($\cos^{-1}[(\frac{\nu}{\nu+1})^{1/2}]$, $\cos^{-1}[(\frac{\nu}{\nu+1})^{1/2}]$) when $\nu \geqslant 2$.

5.4 Design of 2D two-color multigrid methods

5.4.1 Rearrangement of the smoothing order

Suppose that we rearrange the smoothing order from $\{ \text{red} \rightarrow \text{black} \}$ to $\{ \text{black} \rightarrow \text{red} \}$ for the two-grid iteration discussed before. In the wavenumber domain, the black/red Gauss-Seidel iteration matrix becomes

$$\hat{S}_{h,b/r}(\theta) = \begin{bmatrix} J^2 & 0 \\ J & 0 \end{bmatrix} \ \theta_x, \theta_y < \frac{\pi}{2}; \quad \begin{bmatrix} \alpha^2 & 0 \\ \alpha & 0 \end{bmatrix}, \ \theta_x \text{ or } \theta_y = \frac{\pi}{2}; \quad 0, \ \theta_x = \theta_y = \frac{\pi}{2} .$$

This indicates that the computational process that follows is determined by the initial values of the red points only. Several facts can be obtained by modifying the derivation in the previous section slightly. The two-grid method with black/red Gauss-Seidel relaxation consists of two processes

$$M_{11}(\theta) : (r_{\mathbf{k}}, -r_{\tilde{\mathbf{k}}})^T \rightarrow (r_{\mathbf{k}}, -r_{\tilde{\mathbf{k}}})^T , \quad M_{21}(\theta) : (r_{\mathbf{k}}, -r_{\tilde{\mathbf{k}}})^T \rightarrow (b_{\mathbf{k}}, -b_{\tilde{\mathbf{k}}})^T .$$

Asymptotically, its rate of convergence is determined by that of the process $M_{11}(\theta)$. In mathematical terms, we have

$$\rho[\ \hat{K}_h^{2h}(\theta)\ \hat{S}_{h,b/r}^{\nu}(\theta)\ ] = \begin{cases} \rho(K_{12}J^{2\nu-1} + K_{11}J^{2\nu}) & \theta_x, \theta_y < \frac{\pi}{2} \\ \alpha^{2\nu} & \theta_x \text{ or } \theta_y = \frac{\pi}{2} \\ 0 & \theta_x = \theta_y = \frac{\pi}{2} \end{cases} .$$

Since $F_{12} = F_{21}^T$, we obtain the equality

$$\rho(K_{12}J^{2\nu-1} + K_{11}J^{2\nu}) = \rho(K_{21}J^{2\nu-1} + K_{22}J^{2\nu}) ,$$

which implies that the spectral radii of the two-grid methods with either the red/black or black/red Gauss-Seidel relaxation are the same.

Motivated by the 1D algorithm (5.2.8), we consider an improved multigrid method whose two-grid iteration operator $M_{h,i}^{2h}$ is of the form

$$M_{h,i}^{2h} = S_{h,b}\, K_{h,i}^{2h}\, S_{h,r} \,, \qquad K_{h,i}^{2h} = I_{h,r} - I_{2h}^{r} L_{2h}^{-1} I_b^{2h} L_{h,b} \,,$$

where $I_{h,r}$ is the identity operator at red points, $L_{h,b}$ is the restriction of the 5-point discretized Laplacian operator to the black points of Ω_h, and I_b^{2h} and I_{2h}^{r} are the black-to-coarse restriction and coarse-to-red interpolation operators defined by,

$$I_b^{2h} : \left| \begin{matrix} 0 & \frac{1}{8} & 0 \\ \frac{1}{8} & 0 & \frac{1}{8} \\ 0 & \frac{1}{8} & 0 \end{matrix} \right|_b^{2h} , \quad \text{and} \quad I_{2h}^{r} : \left| \begin{matrix} \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & 1 & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} \end{matrix} \right|_{2h}^{r} . \qquad (1)$$

Comparing (5.3.3) and (1), we see that the simplified restriction operator I_b^{2h} and interpolation operator I_{2h}^{r} are in fact obtained respectively by setting the coefficients of the red points of the full weighting operator I_h^{2h} and of the black points of the linear interpolation operator I_{2h}^{h} equal to 0. This change is motivated by the observation that

$$S_{h,b}\, K_h^{2h}\, S_{h,r} = S_{h,b}\, K_{h,i}^{2h}\, S_{h,r} \,,$$

since the residues at the red points are zero before the restriction operation and the values at the black points are not used after the interpolation. The corresponding computational algorithm is stated below.

(1) Perform a Jacobi iteration at the red points.

(2) Calculate residues at the black points and average them with the coefficients specified by I_b^{2h} to obtain residue values at the coarse-grid points.

(3) Solve the system of residue equations on the $2h$-grid, and interpolate the coarse-grid solution to the red points according to I_{2h}^{r}, which is then added back to the original values at the red points.

(4) Perform a Jacobi iteration at the black points.

One important feature of the improved method is that it splits one complete iteration $S_{h,r}\ S_{h,b}$ into two separate operations and uses $S_{h,r}$ and $S_{h,b}$ as presmoother and postsmoother respectively. It is this particular arrangement that makes possible the reduction of computational work associated to the use of the simplified restriction and interpolation operators (1). The improved method has the same convergence rate as the conventional method using either red/black or black/red Gauss-Seidel relaxation, since

$$\rho(S_{h,b}\ K_{h,i}^{2h}\ S_{h,r}) = \rho(S_{h,b}\ K_h^{2h}\ S_{h,r}) = \rho(K_h^{2h}\ S_{h,b/r}) = \rho(K_h^{2h}\ S_{h,r/b}) .$$

The generalization of the improved method to $\nu \geqslant 2$ is straightforward. The key is to position $S_{h,r}$ just before the residue restriction step and $S_{h,b}$ just after the solution interpolation step. For example, when $\nu = 2$, the improved two-grid methods can be

$$S_{h,b/r}\ K_{h,i}^{2h}\ S_{h,b/r}\ , \qquad S_{h,b}\ K_{h,i}^{2h}\ S_{h,r}\ S_{h,r/b}\ , \quad \text{or} \quad S_{h,r/b}\ S_{h,b}\ K_{h,i}^{2h}\ S_{h,r}\ .$$

5.4.2 Smoothing with a relaxation parameter

The red/black Gauss-Seidel relaxation method is a special case of the red/black SOR iteration with the relaxation parameter $\omega = 1$. In the context of single-grid iterative methods, an optimal choice of the relaxation parameter usually improves the convergence rate significantly. However, it has been observed empirically that the choice $\omega = 1$ gives the best efficiency for multigrid methods [3][8]. Since every multigrid method uses a certain kind of single-grid solution method as an essential building block, i.e. for the smoother, there exists a close relationship between single-grid and multigrid methods. Hence, an interesting problem consists in analyzing the link existing between the SOR single-grid method and its corresponding multigrid method.

One common way to analyze a multigrid method is to approximate it by an analysis of a two-grid method as shown in Sections 5.2 and 5.3. However, for $\omega \neq 1$, the 4×4 two-grid iteration matrices are of full rank and the resulting analysis requires the determination of the largest value of the spectral radii of 4×4 matrices. Consequently, it is difficult to perform an exact two-grid analysis. The other common simplified analysis is based on the assumption that ideal restriction and interpolation operators are used to partition the wavenumber domain into low and high wavenumber regions [8], where the partitioning is normally done according to whether they can or cannot be represented on the coarse grid. By such a simplification, we can focus on the error smoothing property in the high wavenumber region. Nevertheless, this analysis does not help to understand the multigrid SOR method either.

5.4.2.1 Analysis

A simplified analysis is proposed below to explain how the multigrid red/black SOR method works and to predict the optimal relaxation parameter. Our analysis is motivated by the observation that in a two-grid method errors are reduced by two effects: the smoothing and the coarse-grid correction effects. So, we first partition the wavenumber domain into the smoothing region Θ^s and the coarse-grid correction region Θ^c depending on which effect is dominant. Then, we concentrate on the smoothing effect in the smoothing region, where the coarse-grid correction effect is ignored. This analysis is applied to the equivalent one-color two-grid matrix given by (5.3.11) below.

Step 1: determination of the smoothing region

We require that the coarse-grid correction region should be within the square $0 < \theta_x , \theta_y < \frac{\pi}{2}$. Within such a square, two parameters η and κ,

$$\eta(\theta_x,\theta_y) = [J^{2\nu}]_{11} = \alpha^{2\nu}, \qquad \kappa(\theta_x,\theta_y) = [K_{eq}]_{11} = 1 - \frac{(1+\beta)(\alpha^2-1)}{2\delta},$$

are used to measure the smoothing and the coarse-grid effects of the two-grid method (5.3.11). The choice $\eta = \alpha^{2\nu}$ is obvious. The choice $\kappa = [K_{eq}]_{11}$ is based on the observation that the coupling term $[K_{eq}]_{12}$ is negligible when (θ_x,θ_y) is close to the origin. However, the nonideal characteristics of I_h^{2h} and I_{2h}^h are still preserved by $[K_{eq}]_{11}$ so that there exists a transition between the coarse-grid correction and the smoothing effects. A reasonable choice of smoothing region is

$$\{ (\theta_x,\theta_y) \in \Theta : \theta_x \geqslant \frac{\pi}{2} \text{ or } \theta_y \geqslant \frac{\pi}{2} \text{ or } \eta(\theta_x,\theta_y) \leqslant \kappa(\theta_x,\theta_y), \text{ if } (\theta_x,\theta_y) \in (0,\frac{\pi}{2})^2 \}.$$

Since the contour $\eta(\theta_x,\theta_y) = \kappa(\theta_x,\theta_y)$ cannot be parameterized easily, for simplicity we modify the above choice slightly and define the smoothing region as

$$\Theta^s = \{ (\theta_x,\theta_y) \in \Theta : \alpha^{2\nu} \leqslant \eta_0, \quad \text{where} \quad \eta_0 = \eta(\theta_0,\theta_0) = \kappa(\theta_0,\theta_0) \}.$$

In the above definition, (θ_0,θ_0) is the point along the line $\theta_x = \theta_y$ where the smoothing parameter η and the coarse-grid correction parameter κ have the same value η_0. This value is then used as threshold to specify the smoothing region. For $\theta_x = \theta_y = \theta$,

$$\kappa(\theta,\theta) = 1 - \frac{1}{2}(1+\cos^2\theta), \qquad \eta(\theta,\theta) = \cos^{2\nu}\theta,$$

which are plotted in Figure 5.3. We see that the smoothing region Θ^s increases as ν increases. The partitioning of the smoothing and coarse-grid correction regions in the folded two-color Fourier domain is depicted in Figure 5.4(a).

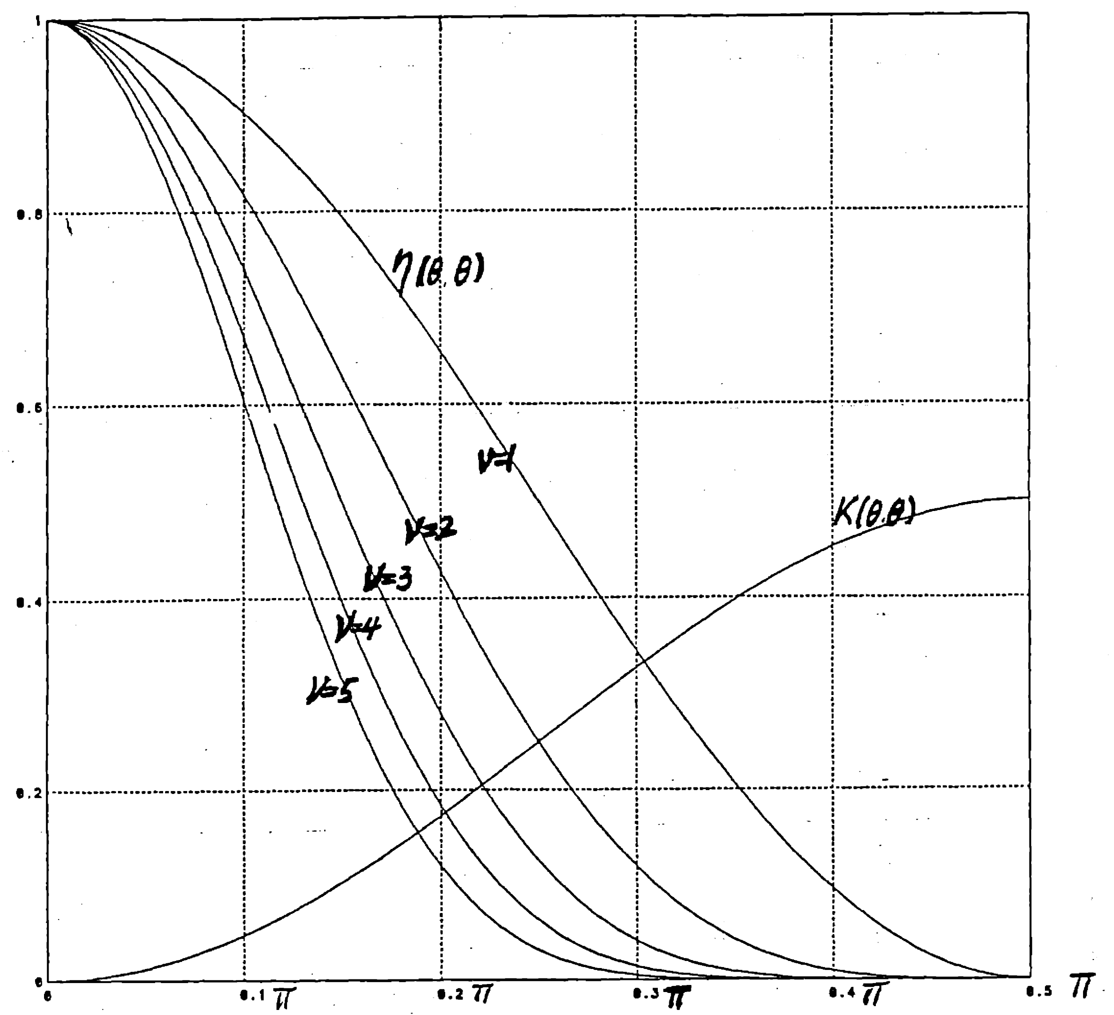


Figure 5.3: Plot of the smoothing parameter $\eta(\theta,\theta)$, $1 \leqslant \nu \leqslant 5$, and of the coarse-grid correction parameter $\kappa(\theta,\theta)$ as functions of θ, $0 < \theta < \frac{\pi}{2}$.

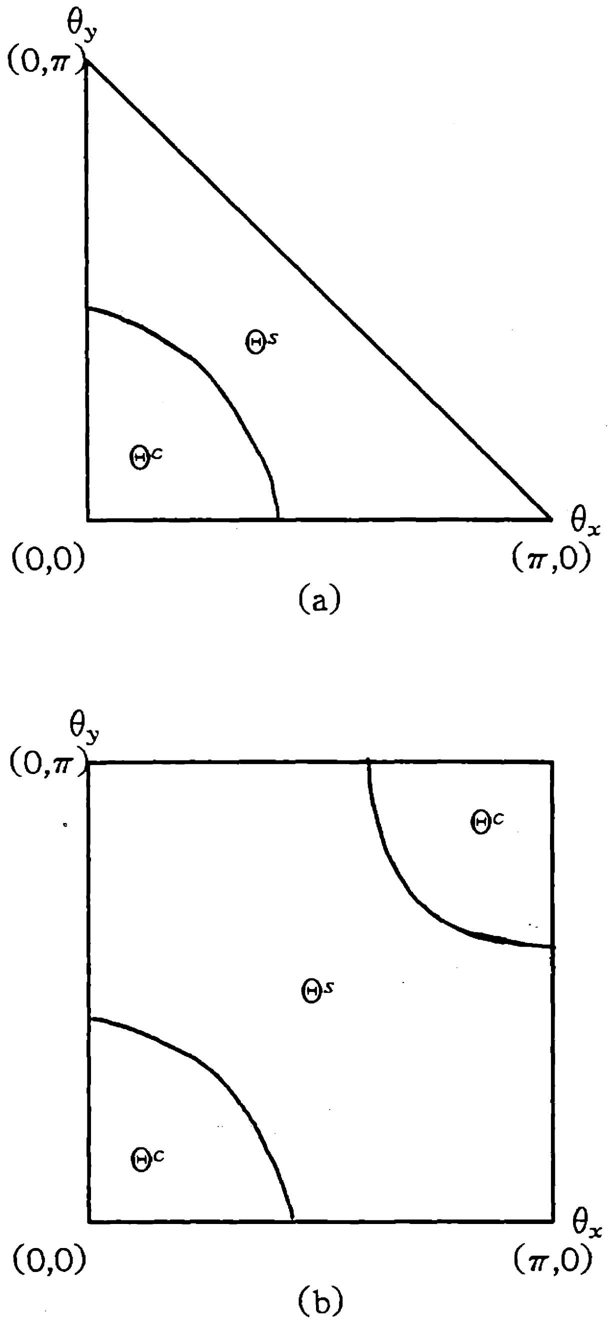


Figure 5.4: Partitioning of the smoothing region and the coarse-grid correction region with $\nu = 2$ in the (a) folded two-color and (b) conventional Fourier domains.

In particular, the single-grid method can be viewed as an extreme case of the two-grid method where ν is so large that the smoothing region covers all feasible discrete wavenumbers. By solving $\kappa(\theta_0,\theta_0) = \eta(\theta_0,\theta_0)$ with $\nu = 1$ and 2, we find that

$$\eta_0 = \eta(\theta_0,\theta_0) = \begin{cases} \frac{1}{3} & \nu = 1 \\ \frac{1}{4} & \nu = 2 \end{cases} .$$

We can also examine the partitioning of the smoothing and coarse-grid regions in the conventional Fourier domain as shown in Figure 5.4(b), where the domain is partitioned into three bands: the low, middle and high wavenumber bands. We see that the red/black Gauss-Seidel relaxation is an effective middle band smoother. The smoothing property becomes poor when (θ_x, θ_y) is close to (0,0) or (π,π) and errors in these regions are primarily solved by the coarse-grid correction operation.

Step 2: design of the SOR smoother in the smoothing region

Once the smoothing region is determined, we adopt a further simplification. It is assumed that within the smoothing region, the coarse-grid correction effect is negligible so that the coarse-grid correction operator is treated as if it were an identity operator. Then, the remaining task is simply to determine the best single-grid SOR method in Θ^s. Since the analysis is standard, we simply summarize the result [9][10]. With respect to $(\hat{r}_{\mathbf{k}}, \hat{b}_{\mathbf{k}})^T$, the red/black SOR iteration matrix is

$$\hat{G}(\theta,\omega) = \hat{S}_{h,b}(\theta,\omega)\,\hat{S}_{h,r}(\theta,\omega) = \begin{bmatrix} 1 & 0 \\ \omega\alpha & 1-\omega \end{bmatrix} \begin{bmatrix} 1-\omega & \omega\alpha \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1-\omega & \omega\alpha \\ (1-\omega)\omega\alpha & \omega^2\alpha^2+1-\omega \end{bmatrix},$$

where $\alpha = \dfrac{\cos\theta_x + \cos\theta_y}{2}$, and $S_{h,r}(\omega)$ and $S_{h,b}(\omega)$ are the damped Jacobi relaxation operators with parameter ω at red and black points. The optimal relaxation parameter

ω^* which minimizes $\max_{\theta \in \Theta^s} \rho[\hat{G}(\theta,\omega)]$ is

$$\omega^* = \frac{2}{1 + [1-\alpha_{\max}^2]^{1/2}} , \quad \text{where} \quad \alpha_{\max}^2 = \max_{\theta \in \Theta^s} \alpha^2 = (\eta_0)^{\frac{1}{\nu}} ,$$

and we define

$$\mu \equiv \max_{\theta \in \Theta^s} \rho[\, \hat{G}(\theta,\omega^*) \,] = \omega^* - 1 ,$$

as the smoothing rate. In particular, when $\nu = 1$ or 2, we find that the predicted optimal relaxation parameter and smoothing rate are given by

$$\omega_{pre}^* = \begin{cases} 1.101 & \nu = 1 \\ 1.171 & \nu = 2 \end{cases} . \tag{2}$$

$$\mu_{pre} = \begin{cases} 0.101 & \nu = 1 \\ 0.171 & \nu = 2 \end{cases} .$$

The predicted convergence rate of the multigrid method is usually related to the smoothing rate via $\rho_{pre} = \mu_{pre}^{\nu}$ [8], so that we have

$$\rho_{pre} = \begin{cases} 0.101 & \nu = 1 \\ 0.029 & \nu = 2 \end{cases} . \tag{3}$$

For larger ν, we have a larger smoothing region as well as a larger optimal relaxation parameter ω^*. When $\Theta^s \approx \Theta$, the two-grid method behaves like a single-grid SOR method and, hence, an optimal relaxation parameter close to 2 is then needed.

5.4.2.2 Numerical results

To study the convergence rate of the V-cycle multigrid method with the red/black SOR smoother, we consider the discretized Laplacian equation on the unit square with zero boundary conditions and $h_x = h_y = \frac{1}{64}$, which has zero as its exact solution.

Although the optimal relaxation parameter ω^* may vary slightly for different problems, we observe that

$$\omega^*_{\exp} \approx \begin{cases} 1.10 & (\nu_1,\nu_2) = (1,0) \\ 1.15 & (\nu_1,\nu_2) = (1,1) \end{cases} , \tag{4}$$

generally gives the optimal or nearly optimal convergence rate. The convergence history for initial guesses corresponding to

(1) a random 2D sequence, and

(2) a smooth function, i.e. $x(x-1)\sin(\pi y)$,

are plotted in Figures 5.5 and 5.6. The y-axis is the 2-norm of the error and the x-axis is the number of V-cycle multigrid iteration. The observed convergence rates are

$$\rho_{\exp} \approx \begin{cases} 0.31 & \omega = 1.00 \\ 0.25 & \omega = 1.10 \end{cases} , \qquad (\nu_1,\nu_2) = (1,0) , \tag{5a}$$

and

$$\rho_{\exp} \approx \begin{cases} 0.12 & \omega = 1.00 \\ 0.03 & \omega = 1.15 \end{cases} , \qquad (\nu_1,\nu_2) = (1,1) . \tag{5b}$$

From (2) - (5), we see that the simple analysis presented in the previous section matches the experimental results closely for $\nu = 2$. However, for $\nu = 1$, there exists a significant discrepancy between our theoretical analysis and the observed convergence rate.

We can conclude that as far as the convergence rate is concerned, the V-cycle multigrid method with $(\nu_1,\nu_2) = (1,1)$ and $\omega^* = 1.15$ usually gives the best result. Thus, if we are interested in solving a system of algebraic equations to a very high order of accuracy, it may be worthwhile to use the red/black SOR smoother in a multigrid algorithm. However, our objective is often to solve a system of equations which is used to approximate a partial differential equation. Since the discretization error for the 5-point discretized Laplacian is $O(h^2)$, one or two cycles of a multigrid iteration are sufficient to achieve this accuracy. In such a case, the improvement of the convergence rate may not offset the additional computational work required by the introduction of a relaxation parameter different from 1. It is also important to emphasize that the full-weighting restriction and the linear interpolation operators given by (5.3.3), rather than the simplified restriction and interpolation operators specified by (1), have to be used when $\omega \neq 1$.

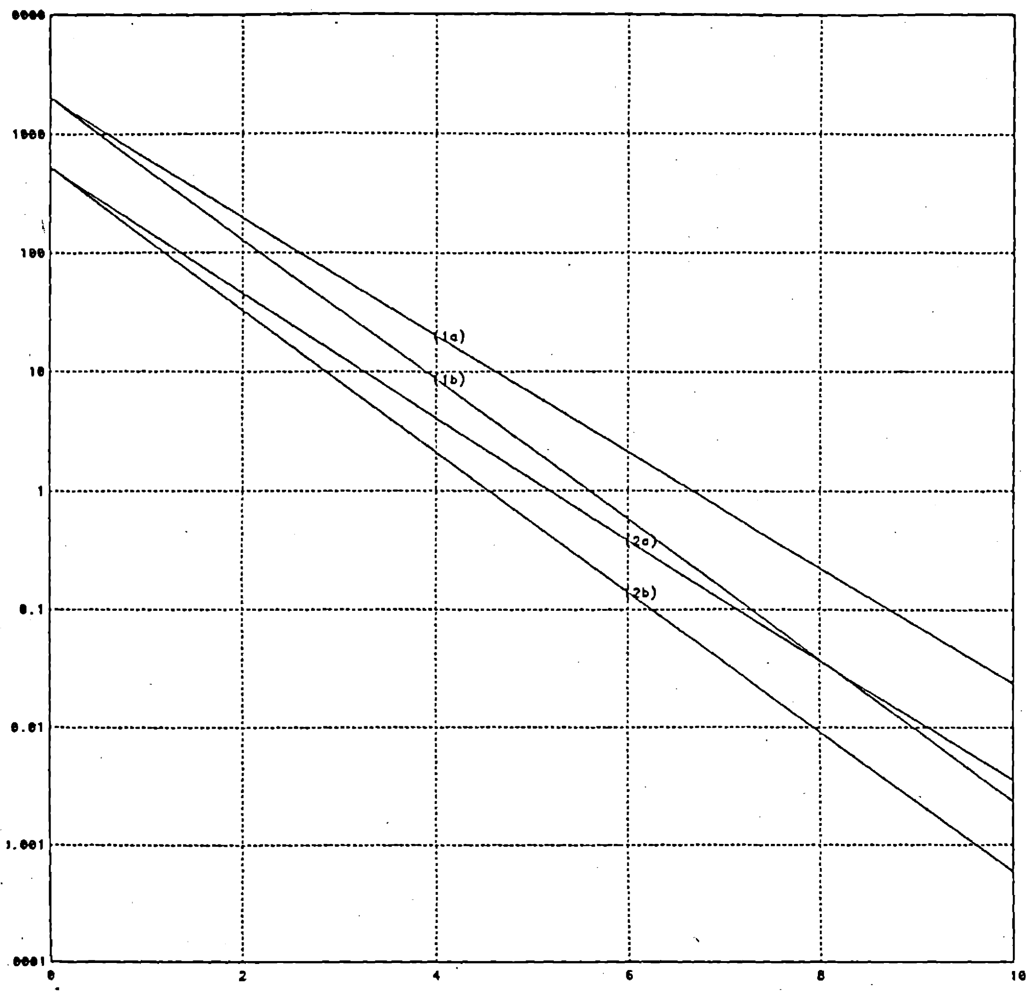


Figure 5.5: Convergence history of the V-cycle multigrid method, where $(\nu_1,\nu_2) = (1,0)$, for two test examples with (a) $\omega = 1$ and (b) $\omega = 1.1$. The x-axis is the number of cycles and the y-axis is the 2-norm of the error.

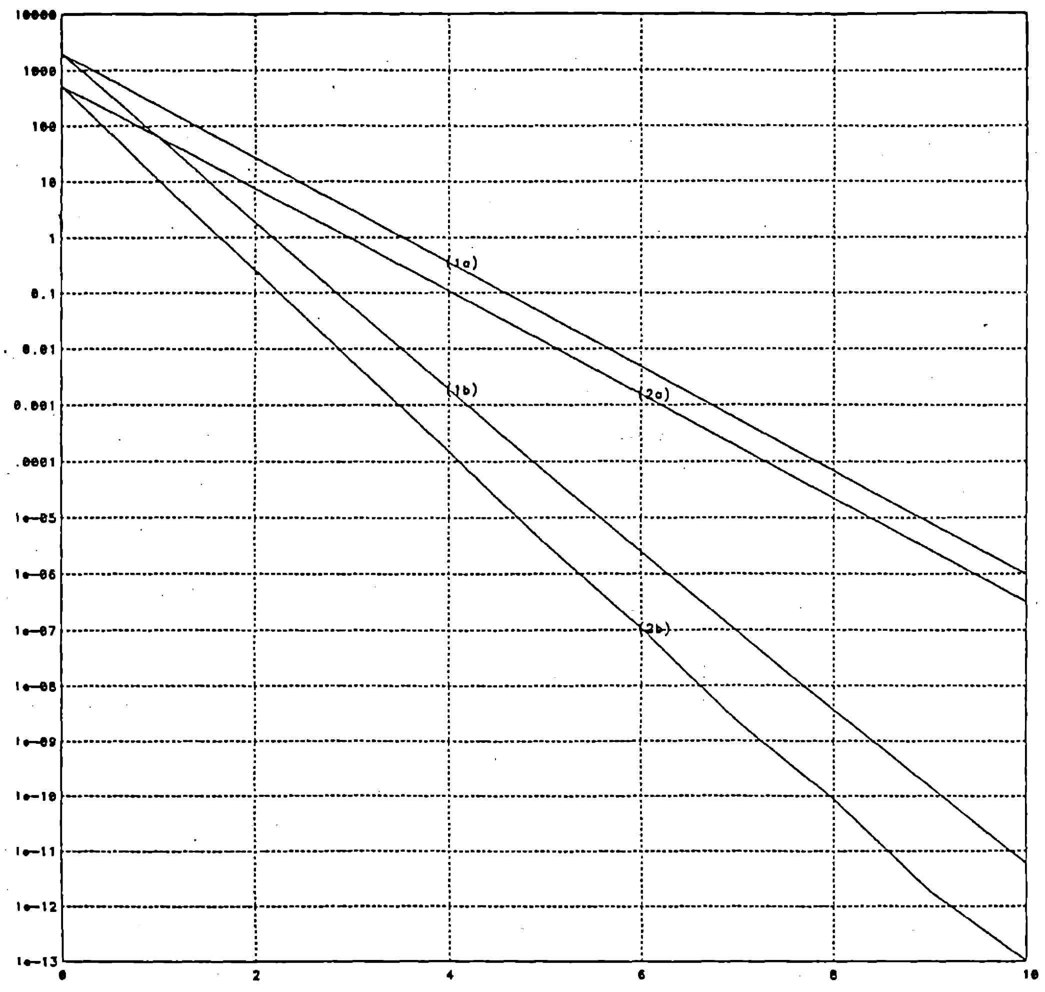


Figure 5.6: Convergence history of the V-cycle multigrid method, where $(\nu_1,\nu_2) = (1,1)$, for two test examples with (a) $\omega = 1$ and (b) $\omega = 1.15$. The x-axis is the number of cycles and the y-axis is the 2-norm of the error.

5.5 Conclusions and extensions

A two-color Fourier analysis method has been proposed to analyze and design multigrid algorithms which employ the red/black Gauss-Seidel iteration. By this analysis, we can clearly explain the coupling phenomenon existing between the low and high wavenumber components of the solution and give an more intuitive derivation of the two-grid analysis for the model Poisson problem. The same analytical approach can also be conveniently applied to the MGR-CH (Multigrid Reduction with checkered Gauss-Seidel relaxations) method [5][7].

We also used a simplified analysis to predict the optimal relaxation parameter for a multigrid SOR method. The improvement of the multigrid convergence rate has been demonstrated both analytically and experimentally. Although multigrid SOR algorithms do not present any advantage for the 5-point stencil discretization of the model Poisson problem, they are expected to be useful for solving systems of equations obtained from high-order discretization schemes such as the 9-point discretized Laplacian operator. Various single-grid SOR schemes have been investigated for this problem already [1] (also see Chapter 4 of this thesis). It is an interesting future research topic to incorporate these single-grid SOR smoothers inside multigrid procedures.

5.6 References

[1] L. Adams, R. J. LeVeque, and D. M. Young, "Analysis of the SOR Iteration for the 9-Point Laplacian," Submitted to *SIAM J. Num. Analy.*.

[2] A. Brandt, "Multigrid Solvers on Parallel Computers," in *Elliptic Problem Solvers*, ed. M. H. Schultz, New York, N.Y.: Academic Press, Inc., 1981, pp. 39-83.

[3] A. Brandt, "Guide to Multigrid Development," in *Multigrid Methods*, ed. W. Hackbusch and U. Trottenberg, New York, N.Y.: Springer-Verlag, 1982, pp. 220-312.

[4] R. E. Crochiere and L. R. Rabiner, *Multirate Digital Signal Processing*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1983.

[5] H. Foerster, K. Stuben, and U. Trottenberg, "Nonstandard Multigrid Techniques Using Checkered Relaxation and Intermediate Girds," in *Elliptic Problem Solvers*, ed. M. Schultz, New York, N.Y.: Academic Press, Inc., 1981, pp. 285-300.

[6] W. Hackbusch, *Multi-Grid Methods and Applications*. Berlin, Germany: Springer-Verlag, 1985.

[7] M. Ries, U. Trottenberg, and G. Winter, "A Note on MGR Methods," *Linear Algebra and Its Application*, vol. 49, pp. 1-26, 1983.

[8] K. Stuben and U. Trottenberg, "Multigrid Methods : Fundamental Algorithms, Model Problem Analysis, and Applications," in *Multigrid Methods*, ed. W. Hackbusch and U. Trottenberg, New York, N.Y.: Springer-Verlag, 1982, pp. 1-176.

[9] R. S. Varga, *Matrix Iterative Analysis*. Englewood Cliffs, N.J. : Prentice-Hall, Inc. , 1962.

[10] D. M. Young, *Iterative Solution of Large Linear Systems*. New York, N.Y.: Academic Press, Inc., 1971.

PART IV : PARALLEL PROCESSING

The last part of this thesis examines the implementation of various single-grid and multigrid solution methods on multiprocessor arrays.

We first study the parallel implementation single-grid solution methods on a mesh-connected processor array. Consider a mesh-connected processor array consisting of $O(N)$ processors used to solve a discretized elliptic PDE with $O(N)$ unknowns. We show that the Jacobi, Gauss-Seidel, SOR, and Conjugate Gradient (CG) methods all require $O(N)$ computation time while the local relaxation method presented in Chapter 3 only requires $O(\sqrt{N})$ computation time. The advantage of the local relaxation method is that it can achieve an acceleration effect with only local communications at each iteration.

Then, we give a brief survey of current research work on parallel implementations of single-grid and multigrid solution methods on multiprocessor arrays.

Chapter 6 : Parallel Implementations of Single-grid and Multigrid Methods on Multiprocessor Arrays

6.1 Introduction

The implementation of a given algorithm on a tightly-coupled multiprocessor machine, can usually be accomplished in two steps, as shown in Figure 6.1. The first step consists in implementing the algorithm on a *virtual* machine whose architecture supports the algorithm in the most natural way. Then, in the second step, the virtual machine is mapped into a specific *physical* machine. The first step requires an analysis of the parallelizability of the algorithm that we consider, and in particular a study of the computation and communication requirements of this algorithm. The second step focuses primarily on the topological structures of the virtual and physical multiprocessor machines, and on their relation.

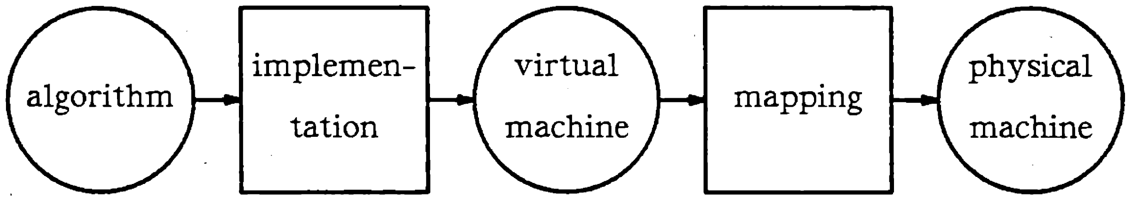


Figure 6.1 : Parallel implementation of algorithms

As discussed in Chapter 1, if the interconnection of the physical machine is chosen to be exactly the same as that of the virtual machine, the mapping problem becomes trivial and the resulting special-purpose architecture is especially efficient for the particular algorithm under consideration. Nevertheless, for most cases, the machine available is not identical with the virtual machine desired. So, the mapping (or embedding) problem is often an interesting and important issue [3][4]. However, it is quite different from the algorithm implementation problem, and it is therefore preferable to treat these two problems separately.

This chapter has two objectives. The first objective is to study the parallel implementation of relaxation algorithms for a class of elliptic PDEs given by (3.2.1) on a 2D domain with $\sqrt{N} \times \sqrt{N}$ grid points on a $\sqrt{N} \times \sqrt{N}$ mesh-connected array. Section 6.2 shows that the performance of the local relaxation method is better than that of other single-grid iterative methods with respect to this implementation. Let the sum of the communication and computation times in one iteration be the processing time per iteration. We find that the implementation of the Jacobi and Gauss-Seidel relaxation schemes requires $O(1)$ processing time per iteration and $O(N)$ iterations to achieve convergence, and that the implementation of the conventional SOR and conjugate gradient (CG) methods requires $O(\sqrt{N})$ processing time per iteration and $O(\sqrt{N})$ iterations to achieve convergence. As a consequence, these various parallel implementations have all a time complexity proportional to $O(N)$. In contrast, the local relaxation algorithm developed in Chapter 3 requires $O(1)$ processing time per iteration and $O(\sqrt{N})$ iterations and, therefore, its time complexity is proportional to $O(\sqrt{N})$ only. The second objective is to review research work concerning the parallel implementation of single-grid and multigrid methods on multiprocessor arrays, which will be presented in Section 6.3. Finally, some concluding remarks are given in Section 6.4.

6.2 Implementation of Single-grid Methods on Mesh-connected Arrays

Consider an elliptic PDE problem on a square domain discretized by a finite-difference method on a grid Ω_h with $\sqrt{N} \times \sqrt{N}$ grid points. The most natural supporting architecture for solving this problem is a $\sqrt{N} \times \sqrt{N}$ mesh-connected array. This is due to the following observations. First, there is a one-to-one correspondence between grid points of the discretization grid Ω_h and processors in the array and the mapping between processors and grid points is trivial. Second, processors are connected in the same way as discretization points are related to each other, so that the exchange of data which is required by the algorithm can often, but not always, be easily implemented as a local communication exchange between neighboring processors. Hence, by assigning one processor to update the value of the solution at one grid point, we obtain a natural parallel computational scheme to solve the discretized PDE problem. In fact, this idea simply exploits the parallelizability of the PDE problem in the space domain.

The total running time for an iterative algorithm equals the product of the processing time per iteration and the number of iterations. In a sequential machine, the processing time per iteration is determined by operation counts, especially by the number of floating point operations required. In a multiprocessor machine, however, the processing time per iteration depends heavily on the communication scheme required by the algorithm chosen. Algorithms using only local communication will take $O(1)$ communication time, while those using global communication require $O(\sqrt{N})$ communication time per iteration since for an array with N processors, communications between processors located on opposite sides of the array will take

$O(\sqrt{N})$ time. We thus seek algorithms with fast convergence rate, short computation time, and primarily local communication.

Starting from equation (3.2.2), we can discuss the details of implementing different iterative algorithms with a mesh-connected processor array. Let us label the processor of a $\sqrt{N} \times \sqrt{N}$ array by a 2D index, and assign the processor at coordinate (n_x, n_y) the responsibility of calculating the value of u_{n_x,n_y}. Direct communication is allowed only between neighboring processors. At iteration $m+1$, each processor may combine the estimated value of u^m in neighboring processors, together with its own estimate of $u^m_{n_x,n_y}$, in order to develop a new estimate $u^{m+1}_{n_x,n_y}$.

For the Jacobi method, we have

$$u^{m+1}_{n_x,n_y} = d^{-1}_{n_x,n_y} (l_{n_x,n_y} u^m_{n_x-1,n_y} + r_{n_x,n_y} u^m_{n_x+1,n_y} + b_{n_x,n_y} u^m_{n_x,n_y-1} + t_{n_x,n_y} u^m_{n_x,n_y+1} + s_{n_x,n_y}) .$$

According to the above iterative equation, each processor uses the values of u^m obtained by its nearest neighbors to update its value at the current iteration. Processing time per iteration is constant, because both communication time and computation time are constant.

If the grid point (n_x, n_y) is called a red point when $n_x + n_y$ is even, and a black point when $n_x + n_y$ is odd, the Jacobi method can be viewed in space and time as consisting of two interleaved, and totally independent *computational waves* alternating between red and black points. This phenomenon is illustrated in Figure 6.2, where the one dimensional grid with red/black partitioning is shown in the horizontal direction while the evolution from one iteration to the next is indicated in the vertical direction. The solid and dotted lines represent two value-updating processes evolving with time, or two computational waves.

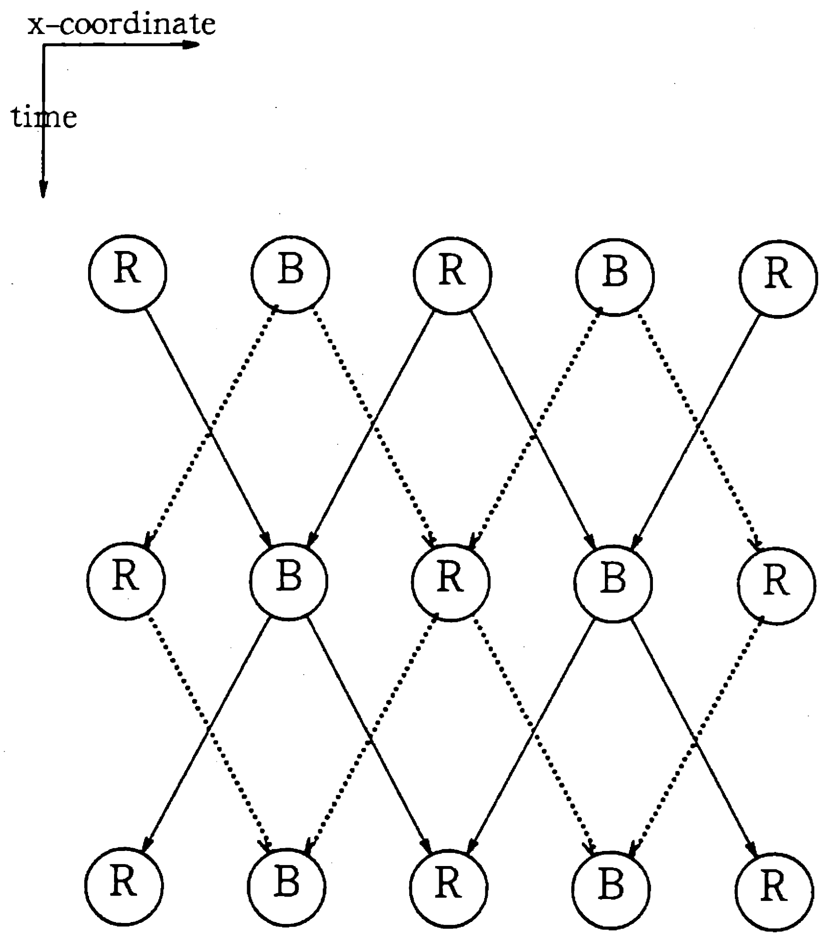


Figure 6.2: 1D Jacobi relaxation with red/black partitioning

In fact, these two waves result in unnecessary redundancy. We need only one wave to get the answer, since both waves converge to the same final values. If we delete one computational wave, the rate of utilization of the processors becomes one half, i.e., every processor works only half of the time. Therefore, we may group one red point and one black point together and assign them to a single processor. This saves half of the hardware cost without loss of computational efficiency (See Figure 6.3).

For the Gauss-Seidel relaxation with red/black point partitioning. we have the local equations,

red points ($n_x + n_y$ is even) :

$$u_{n_x,n_y}^{m+1} = d_{n_x,n_y}^{-1} \left(l_{n_x,n_y} u_{n_x-1,n_y}^{m} + r_{n_x,n_y} u_{n_x+1,n_y}^{m} + b_{n_x,n_y} u_{n_x,n_y-1}^{m} + t_{n_x,n_y} u_{n_x,n_y+1}^{m} + s_{n_x,n_y} \right),$$

black points ($n_x + n_y$ is odd) :

$$u_{n_x,n_y}^{m+1} = d_{n_x,n_y}^{-1} \left(l_{n_x,n_y} u_{n_x-1,n_y}^{m+1} + r_{n_x,n_y} u_{n_x+1,n_y}^{m+1} + b_{n_x,n_y} u_{n_x,n_y-1}^{m+1} + t_{n_x,n_y} u_{n_x,n_y+1}^{m+1} + s_{n_x,n_y} \right).$$

Other partitionings will lead to different Gauss-Seidel schemes; however, the red/black partitioning approach is preferred for parallel implementation on mesh-connected arrays, because of its efficiency and simplicity. For the case of a one-dimensional grid, we find that the Gauss-Seidel iteration is equivalent to the computational wave of the Jacobi iteration shown by the dotted line in Figure 6.2. Therefore, we can save one half of the computational work by using Gauss-Seidel iteration on either a single processor or a mesh-connected array.

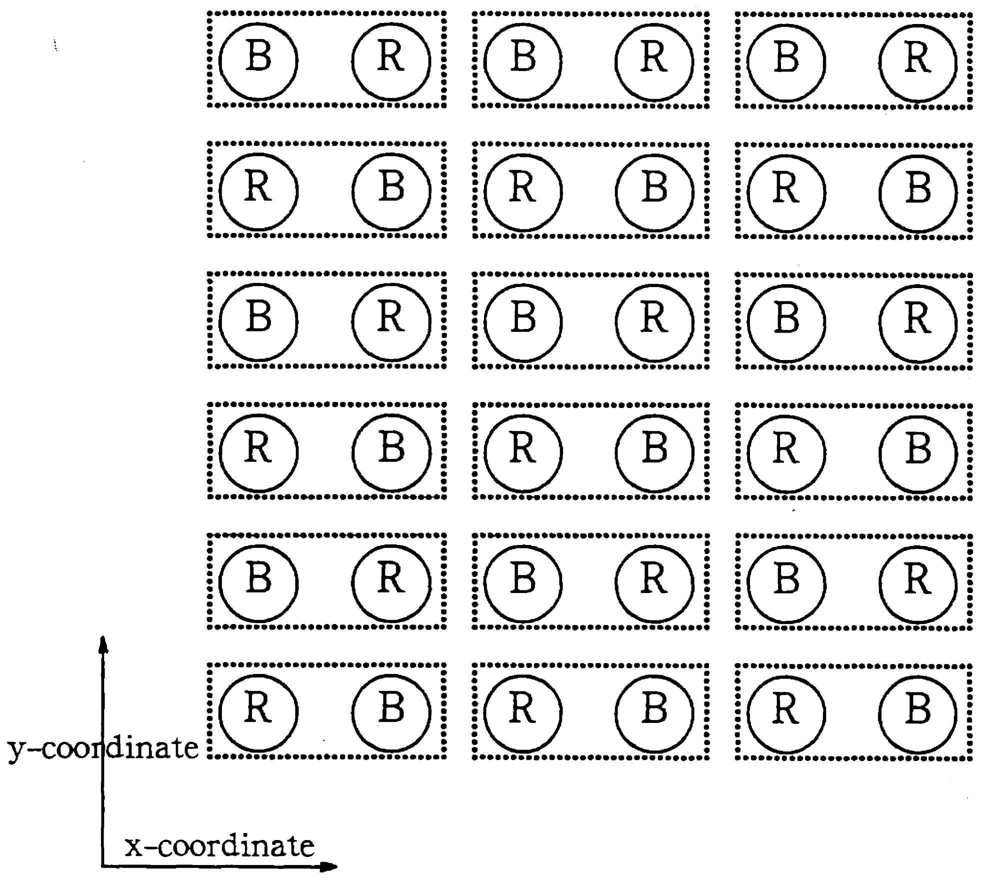


Figure 6.3: 2D red/black partitioning and grouping

As mentioned in Chapter 3, the main limitation of the Jacobi or Gauss-Seidel iterative methods is their slow convergence rate. The number of iterations needed to achieve the accuracy of the discretization error $O(h^2)$ for the model Poisson problem is proportional to $O(N)$ [16][31]. Since the processing time per iteration is constant, the total running time is also proportional to $O(N)$.

By appropriately selecting relaxation parameters for accelerated relaxation algorithms such as the Chebyshev semi-iterative (CSI) method and the successive over-relaxation (SOR) method, these algorithms converge faster. For a given mesh-connected processor array, if we know these quantities a priori and broadcast them to all processors in the loading stage, each processor can compute the acceleration parameters on its own without additional communication cost. In this case, although the accelerated schemes require a little more computation and memory than the basic Jacobi and Gauss-Seidel relaxation schemes, they present some significant advantages. The reason is that the number of iterations needed is reduced tremendously, becoming $O(\sqrt{N})$ for the model Poisson problem for both acceleration schemes [16]. However, in general we do not know the eigenvalues of the basic relaxation matrix in advance and have to estimate them by some adaptive procedure. Since all the estimation procedures developed until now require the computation of the norms of some global vectors, global communication cannot be avoided. This means that the communication cost for a single iteration in a mesh-connected array becomes $O(\sqrt{N})$. As a consequence, the processing time per iteration is $O(\sqrt{N})$ and the total running time becomes $O(N)$ again.

Comparing this result with the result obtained for basic relaxation methods, it seems that we do not benefit from acceleration schemes when we seek to implement iterative algorithms in parallel on mesh-connected arrays. This can be easily explained by noting that for a single processor, there is no distinction between local and global communications, since all data are fetched from the same memory, while for a mesh-connected processor array, long range communication costs much more than short range communication. Any time saving due to the acceleration in these schemes is canceled out by the global communication required.

In addition to the above relaxation algorithms, another important class of algorithms for solving systems of linear equations can be derived from an optimization principle. The conjugate gradient (CG) algorithm is an example [16]. Without considering rounding errors, a theoretical analysis indicates that the CG algorithm is able to solve the discretized PDE exactly in $O(N)$ steps, using only $O(N^2)$ computation and $O(N)$ storage on a single processor. In practice, experience shows that the CG method, when applied to the PDE problem, usually converges in $O(\sqrt{N})$ steps even with rounding errors. Unfortunately, on a mesh-connected array, this algorithm is slowed by the need to compute several inner products of $O(N)$ length vector. Computing the inner product of two vectors whose entries are distributed over a mesh-connected array requires global communication. We therefore encounter the same difficulties as for the accelerated relaxation methods.

The local relaxation method presented in Chapter 3 is a computational algorithm suitable for parallel implementation on mesh-connected processor arrays, because it has the same acceleration effect as SOR and uses only local communication. For example, let us repeat equations (3.2.5)-(3.2.7) as follows, red points ($n_x + n_y$ is even) :

$$u^{m+1}_{n_x,n_y} = (1 - \omega_{n_x,n_y})\, u^{m}_{n_x,n_y} \tag{1a}$$

$$+ \omega_{n_x,n_y}\, d^{-1}_{n_x,n_y}\, (l_{n_x,n_y}\, u^{m}_{n_x-1,n_y} + r_{n_x,n_y}\, u^{m}_{n_x+1,n_y} + b_{n_x,n_y}\, u^{m}_{n_x,n_y-1} + t_{n_x,n_y}\, u^{m}_{n_x,n_y+1} + s_{n_x,n_y}),$$

black points ($n_x + n_y$ is odd) :

$$u^{m+1}_{n_x,n_y} = (1 - \omega_{n_x,n_y})\, u^{m}_{n_x,n_y} \tag{1b}$$

$$+ \omega_{n_x,n_y}\, d^{-1}_{n_x,n_y}\, (l_{n_x,n_y}\, u^{m+1}_{n_x-1,n_y} + r_{n_x,n_y}\, u^{m+1}_{n_x+1,n_y} + b_{n_x,n_y}\, u^{m+1}_{n_x,n_y-1} + t_{n_x,n_y}\, u^{m+1}_{n_x,n_y+1} + s_{n_x,n_y}),$$

where

$$\omega_{n_x,n_y} = \frac{2}{1 + (1 - \rho^2_{n_x,n_y})^{1/2}}, \tag{2}$$

and where

$$\rho_{n_x,n_y} = \frac{2}{d_{n_x,n_y}} [\, (l_{n_x,n_y}\, r_{n_x,n_y})^{1/2} \cos\frac{\pi}{N_x+1} + (t_{n_x,n_y}\, b_{n_x,n_y})^{1/2} \cos\frac{\pi}{N_y+1}\,]. \tag{3}$$

The implementation of the above local relaxation algorithm is straightforward. It is easy to see that as long as we know the size of the grid, i.e., N_x and N_y, we can broadcast this information to all processors in the loading stage. Each processor has to compute its own relaxation parameters once according to equations (2) and (3); then the local iterations specified by equation (1) can be performed in parallel for all processors with only local communication. Since the local relaxation method uses local communication, the computation time per iteration is $O(1)$. We showed that the number of iterations for typical test problems is proportional to $O(\sqrt{N})$ in Section 3.6. Therefore, the total running time becomes $O(\sqrt{N})$.

For a $\sqrt{N} \times \sqrt{N}$ processor array, the constraint that each processor should contain a minimum amount of global information implies that the lower bound for the computation time for any algorithm is $O(\sqrt{N})$, since it takes $O(\sqrt{N})$ time for the data at one edge of the array to move to the opposite edge. It turns out that the local relaxation method achieves this lower bound. The above discussion is summarized in Table 6.1.

Iterative Methods	Time per Iteration	Number of Iterations	Total Time Needed
Jacobi	$O(1)$	$O(N)$	$O(N)$
G-S	$O(1)$	$O(N)$	$O(N)$
SOR	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(N)$
CG	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(N)$
LR	$O(1)$	$O(\sqrt{N})$	$O(\sqrt{N})$

Table 6.1 : Performance comparison of various single-grid algorithms implemented on a $\sqrt{N} \times \sqrt{N}$ mesh-connected processor array for typical test problems.

Although the convergence and convergence rate properties of a local relaxation version of the 2-level 4-color SOR method discussed in Chapter 4 have yet to be examined, there is no foreseeable fundamental difficulty in this generalization. Suppose this generalization is possible. It is easy to see that the parallel implementation of the 2-level 4-color local relaxation method is straightforward (cf. Figures 4.2 and 4.3). Each processor needs to compute the local block and point relaxation parameters, ω_{b,n_1,n_2} and ω_{p,n_1,n_2} based on the boundary conditions and the coefficients of the local difference equation. Then, since all data exchange occurs between neighboring processors, the communication cost per iteration is $O(1)$. We have already shown that the number of iterations of the 2-level SOR scheme for the Poisson problem is proportional to $O(\sqrt{N})$. Hence, the total running time is proportional to $O(\sqrt{N})$.

6.3 Related Research Work

As mentioned in Chapter 1, one way to speed up single-grid algorithms is to vectorize operations and to pipeline them by using a vector supercomputer such as a Cray-1 [19] or an array processor [29][18]. This computational scheme is basically a uniprocessor approach, and will not be addressed here.

There has been a number of studies on parallel implementations of single-grid and multigrid algorithms for solving elliptic PDEs in a multiprocessor computing environment. Different algorithms, architectures, or issues were examined by different researchers, which are briefly reviewed in this section.

(1) Parallel Multicolor SOR Solvers

Work along this direction has been done by Evans [10], Adams and Ortega [1], O'Leary [23], and Adams and Jordan [2]. In their work, several issues were investigated under the assumption that a global relaxation parameter was available.

Evans examined the parallel implementation of the red/black SOR method for solving the Laplace equation with a small number of processors [10]. He grouped 2×1, 2×2 or 3×3 points together as a block, and used 2, 4 or 9 processors to update values of points within a block by a red/black SOR method. In fact, the basic idea consists merely in using a small processor array to operate in a blockwise fashion on grid points of a single-grid.

In Adams and Ortega's work [1], a multicolor SOR method was introduced in which points of the same color are not coupled through the finite-difference discretization scheme. For example, a 5-point stencil discretization requires 2 colors (red and black), a 9-point stencil discretization requires 4 colors (red, orange, black and blue), and so on. Since values at points of the same color do not depend on each other due to the nature of the coloring scheme, they can be updated simultaneously. If there are N grid points, the maximal parallelism can be achieved by using $\frac{N}{p}$ processors for a p-color SOR scheme.

A more interesting problem is the analysis of the convergence of various SOR schemes using different orderings or colorings [2][22][23]. Although more parallelism can be achieved by using the multicolor SOR scheme with different orderings, it is important to know whether the convergence rate of these schemes remains the same as that of the rowwise (or columnwise) ordered SOR scheme. A data flow or a tilted grid concept was used to clarify this issue [2][23]. It appears that the multicolor SOR scheme is usually as efficient as the SOR scheme using the rowwise ordering.

In contrast with the work mentioned above, work in this thesis focuses on the selection of optimal relaxation parameters for various multicolor SOR schemes.

(2) Parallel Domain Decomposition Solvers

Numerical elliptic PDE algorithms relying on domain decomposition techniques [8][20] are also easy to parallelize. Roughly speaking, these methods regard the domain of a given problem as a union of several subdomains so that the problem in each subdomain can be easily solved. These methods are particularly attractive for PDEs with irregular domains, or with nonuniform properties in different regions.

The scheme considered by Chan and Resasco consists in dividing a rectangle into parallel strips, performing uncoupled fast Poisson solvers on each subdomain and, then, computing the interface variables by fast Fourier transform [8]. It is suitable for parallel implementation since problems in all subdomains can be solved in parallel and since the only information that needs to be exchanged concerns interface variables. Several preconditioned conjugate gradient-based domain decomposition techniques implemented on a hypercube were compared by Keyes and Gropp [20].

(3) Parallel Multigrid Solvers

Since multigrid methods provide the most efficient way of solving elliptic PDEs, the implementation of multigrid algorithms on multiprocessor arrays has received a large amount attention recently [5][7][9][12][15][17][21].

The most natural architecture for implementing multigrid algorithms is a pyramidal processor array [12][21][30]. Nevertheless, it has been shown that if we assign one processor to each grid point, this design cannot avoid a loss of efficiency as the grid size tends to zero [7]. There are two possible ways to limit the loss of efficiency. The first is to design some kind of grouping scheme which maps a certain number of grid points to a single processor. The second is to design concurrent multigrid algorithms which operate at every grid level simultaneously [21]. Details of these two ideas remain to be pursued.

The mapping of multigrid algorithms on other types of processor arrays such as mesh-connected arrays [5][15][17] and hypercubes [9], has also been studied.

(4) Other Parallel Elliptic Solvers

In addition to the previous three classes of algorithms, parallel implementations of other single-grid algorithms such as the preconditioned conjugate gradient (PCG) method [26], the alternating direction implicit (ADI) method [6] and a banded Gaussian elimination method [6][25] in a multiprocessor environment, especially on a hypercube, have also been considered by researchers.

(5) Mapping, Partitioning and Communication Schemes

Other problems related to the parallel implementation of single-grid algorithms in a multiprocessor environment include the mapping from an irregular problem domain into a regular processor domain [4][11][13], the partitioning of grid points among processors [24], and the choice of communication scheme for different architectures [14][27][28]. Since most practical problems have irregular domains and the number of grid points is usually larger than that of processors, the investigation of these problems is important in practice.

6.4 Conclusions and Extensions

In this chapter, we have compared parallel implementations of several iterative methods on a mesh-connected processor array, and showed that the local relaxation method is particularly attractive since it only requires local communication and has a very fast convergence rate. There are problems which should be investigated more carefully in the future. One question is how to terminate the local relaxation method. A centralized termination scheme can work as follows. The residues of the local processors are pipelined to a certain processor, say, the central one. Then this processor determines the termination time of the relaxation algorithm and broadcasts a "stop" signal to all processors. The above procedure can be performed concurrently with the local relaxation procedure. Note however that because we avoid the use of global communications in the termination procedure by pipelining the local residues which are sent to the central processor, this processor uses the error residues of the local processors at different iterations in determining whether the algorithm should be stopped. A distributed termination scheme would also be useful.

We have also reviewed current research work on the parallel implementation of numerical PDE algorithms on multiprocessor arrays. This is a rapidly growing field so that it is difficult to give a complete survey. Nevertheless, we have tried to divide current research activities into broad areas and to identify the main trends in this field. In particular, we feel that domain decomposition techniques and parallelized multigrid algorithms are two promising areas for further investigation. Also, although mapping and partitioning problems have already been studied by researchers, the results that have been obtained in this area are rather preliminary, and further studies are needed.

6.5 References

[1] L. Adams and J. M. Ortega, "A Multi-Color SOR Method for Parallel Computation," ICASE report, 82-9, Apr. 1982.

[2] L. Adams and H. F. Jordan, "Is SOR Color-Blind," *SIAM J. Sci. Stat.*, vol. 7, no. 2, Apr. 1986.

[3] S. N. Bhatt and I. C.F. Ipsen, "How to Embed Trees in Hypercube," Research Report YALEU/DCS/RR-443, Dec. 1985.

[4] S. H. Bokhari, "On the Mapping Problem," *IEEE Trans. on Computers*, vol. C-30, no. 3, pp. 207-214, Mar. 1981.

[5] A. Brandt, "Multigrid Solvers on Parallel Computers," in *Elliptic Problem Solvers*, ed. M. H. Schultz, New York, N.Y.: Academic Press, Inc., 1981, pp. 39-83.

[6] T. F. Chan, Y. Saad, and M. H. Schultz, "Solving Elliptic Partial Differential Equations on the Hypercube Multiprocessor," Research Report YALEU/DCS/RR-373, Mar. 1985.

[7] T. F. Chan and R. Schreiber, "Parallel Networks for Multi-Grid Algorithms: Architecture and Complexity," *SIAM J. Sci. Stat. Comput.*, vol. 6, pp. 698-711, Jul. 1985.

[8] T. F. Chan and D. C. Resasco, "A Domain-Decomposed Fast Poisson Solver on a Rectangular," Research Report YALEU/DCS/RR-409, Aug. 1985.

[9] T. F. Chan and Y. Saad, "Multigrid Algorithms on Hypercube Multiprocessor," *IEEE Trans. on Computers*, vol. C-35, no. 11, pp. 969-977, Nov. 1986.

[10] D. J. Evans, "Parallel S.O.R. Iterative Methods," *Parallel Computing*, vol. 1, pp. 3-18, 1984.

[11] D. Gannon, "On Mapping Non-uniform PDE Structures and Algorithms onto Uniform Array Architectures," in *Proceeding of International Conference on Parallel Processing*, 1981.

[12] D. Gannon and J. Van Rosendale, "Highly Parallel Multigrid Solvers for Elliptic PDEs: An Experimental Analysis," ICASE Report 82-36, 1982.

[13] D. Gannon and J. Van Rosendale, "Parallel Architectures for Iterative Methods on Adaptive, Block Structured Grids," in *Elliptic Problem Solvers II*, ed. J. Van Rosendale, New York, N.Y.: Academic Press, Inc., 1984, pp. 93-104.

[14] D. Gannon and J. Van Rosendale, "On the Impact of Communication Complexity in the Design of Parallel Numerical Algorithms," ICASE Report 84-41, Aug. 1984.

[15] D. Gannon and J. Van Rosendale, "On the Structure of Parallelism in a Highly Concurrent PDE Solver," *Journal of Parallel and Distributed Computing*, vol. 3, no. 1, pp. 106-135, 1986.

[16] L. A. Hageman and D. M. Young, *Applied Iterative Methods.* New York, N.Y. : Academic Press, Inc. , 1981.

[17] H.-C. Hoppe and H. Muhlenbein, "Parallel Adaptive Full-multigrid Methods on Message-based Multiprocessors," *Parallel Computing*, vol. 3, pp. 269-287, 1986.

[18] K. Hwang and F. A. Briggs, *Computer Architecture and Parallel Processing.* New York, N.Y.: McGraw-Hill, Inc., 1984.

[19] T. L. Jordan, "A Guide To Parallel Computation and Some Cray-1 Experiences," in *Parallel Computations*, ed. G. Rodrigue, New York, N.Y.: Academic Press, Inc., 1982, pp. 1-50.

[20] D. E. Keyes and W. D. Gropp, "A Comparison of Domain Decomposition Techniques for Elliptic Partial Differential Equations and Their Parallel Implementation," Research Report YALEU/DCS/RR-448, Dec. 1985.

[21] C.-C. Kuo, "Parallel Algorithms and Architectures for Solving Elliptic Partial Differential Equations," Master Thesis, Report LIDS-TH-1432, Laboratory for Information and Decision Systems, MIT, Cambridge, MA, Jan. 1985.

[22] R. J. LeVeque and L. N. Trefethen, "Fourier Analysis of the SOR Iteration," Numerical Analysis Report 86-6, Department of Mathematics, MIT, Cambridge, MA. ICASE Report 86-93, Sep. 1986.

[23] D. P. O'Leary, "Ordering schemes for Parallel Processing of Certain Mesh Problems," *SIAM J. Sci. Stat. Comput.*, vol. 5, no. 3, pp. 620-632, Sep. 1984.

[24] D. A. Reed, L. M. Adams, and M. L. Patrick, "Stencils and Problem Partitionings: Their Influence on the Performance of Multiple Processor Systems," ICASE Report 86-24, May 1986.

[25] Y. Saad and M. H. Schultz, "Parallel Direct Methods for Solving Banded Linear Systems," Research Report YALEU/DCS/RR-387, Aug. 1985.

[26] Y. Saad and M. H. Schultz, "Parallel Implementations of Preconditioned Conjugate Gradient Methods," Research Report YALEU/DCS/RR-425, Oct. 1985.

[27] Y. Saad and M. H. Schultz, "Data Communication in Hypercube," Research Report YALEU/DCS/RR-428, Oct. 1985.

[28] Y. Saad and M. H. Schultz, "Data Communication in Parallel Architectures," Research Report YALEU/DCS/RR-461, Mar. 1986.

[29] M. H. Schultz, "Solving Elliptic Problems on an Array Processor System," in *Elliptic Problem Solvers II*, ed. A. Schoenstadt, New York, N.Y.: Academic Press, Inc., 1984.

[30] L. Uhr, "Pyramid Multi-computer Structures and Augumented Pyramids," in *Computing Structures for Image Processing*, ed. M.J.B. Duff, London: Academic Press, 1983, pp. 95-112.

[31] R. S. Varga, *Matrix Iterative Analysis*. Englewood Cliffs, N.J. : Prentice-Hall, Inc. , 1962.

Chapter 7 : Conclusions and Extensions

7.1 Fourier Transform Domain Approach for Numerical PDEs: A Retrospective

The partial differential equations listed in Table 7.1 below play an important role in numerical analysis, since these equations not only govern many common physical phenomena but can also be used to approximate more complicated equations. As is clear from Table 7.1, the equations that we consider are first- and second-order differential equations involving two variables.

(a) First-order wave equation	$c \frac{\partial u}{\partial x} = \frac{\partial u}{\partial t}$
(b) Second-order wave equation	$c^2 \frac{\partial^2 u}{\partial x^2} = \frac{\partial^2 u}{\partial t^2}$
(c) Heat equation	$\alpha \frac{\partial^2 u}{\partial x^2} = \frac{\partial u}{\partial t}$
(d) Poisson equation	$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f$
(e) Helmholtz equation	$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + k^2 u = 0$
(f) Convection-diffusion equation	$\alpha \frac{\partial^2 u}{\partial x^2} + c \frac{\partial u}{\partial x} = \frac{\partial u}{\partial t}$
(g) Cauchy-Riemann equation	$\left\{ \begin{array}{l} \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = f \\ \frac{\partial u}{\partial y} - \frac{\partial v}{\partial x} = g \end{array} \right.$
(h) Inviscid Burgers equation	$u \frac{\partial u}{\partial x} = \frac{\partial u}{\partial t}$
(i) Burgers equation	$\alpha \frac{\partial^2 u}{\partial x^2} + u \frac{\partial u}{\partial x} = \frac{\partial u}{\partial t}$

Table 7.1: Several model equations with two variables.

An understanding of the essential issues occurring in the numerical solution of these equations helps us to understand the solution of more general problems. The above equations are therefore called *model* equations [2]. The first order wave equation, the heat equation and Poisson equation are the simplest hyperbolic, parabolic and elliptic PDEs respectively. The inviscid Burgers and Burgers equations are nonlinear, and their linearized forms are respectively the first-order wave equation and the convection-diffusion equation.

In Part I of this thesis, finite-difference formulas were derived for discretizing the Poisson, Helmholtz and convection-diffusion equations. In Parts II and III of this thesis, single-grid and multigrid solution methods were designed primarily for solving the Poisson equation. All developments were based on the Fourier transform domain approach. Since equations (a) to (g) are all linear and have constant-coefficients, Fourier analysis and its variations often play an important role in analyzing their discretization and solution.

The Fourier approach has been applied to time-dependent problems for a long time. In the area of finite-difference methods, it has been used for example to study the stability of linear multistep methods for the numerical integration of ODEs [11]. For linear hyperbolic and parabolic PDEs with simple boundary conditions, a common stability test based on Fourier analysis is known as the von Neumann method [16]. This has been generalized to problems with more complicated boundary conditions or interfaces [8][21]. Furthermore, when a numerical wave propagates in a grid according to a finite-difference model, some interesting phenomena such as dispersion, dissipation, and parasitic waves occur. These phenomena can also be easily explained by

Fourier analysis [19][20][23]. In the area of spectral methods, Fourier spectral methods are naturally developed and analyzed by the Fourier approach [5][13]. Since Fourier analysis provides a simple analytical tool, Fourier spectral methods are better understood than other spectral methods such as the Chebyshev spectral methods [9][24].

The application of the Fourier approach to elliptic problems has received much less attention than for time-dependent problems. When comparing different iterative methods for solving elliptic PDEs, a standard test problem is the Poisson equation on the square with Dirichlet boundary conditions, which is therefore called the *model problem* (for elliptic PDEs). The model problem analysis can be viewed as a variation of the Fourier approach, where a set of sine functions is used as the basis. In the area of single-grid methods, the model problem analysis applies to many stationary iterative methods including the Jacobi method, the Gauss-Seidel method, the SOR method, the Chebyshev semi-iterative method [22], alternating direction implicit methods [22], and approximate factorization methods [10][17]. In [4], Chan and Elman applied Fourier analysis to reexamine these methods as well as preconditioners for the Poisson equation with periodic boundary conditions, and compared their results with the classical results based on Dirichlet boundary conditions. It turns out that results obtained by Fourier analysis and the model problem analysis are consistent in many, but not all, cases. In the area of multigrid methods, smoothing is conveniently analyzed by Fourier analysis [3][12]. A two-grid model problem analysis has been presented in [18].

Since this thesis focuses on methods related to the SOR iteration, we give a historical survey concerning the Fourier analysis of this method. For the 5-point stencil case, the earliest model problem analysis was given by Frankel [6]. He derived the optimal relaxation parameter and the convergence rate for the SOR method with natural ordering [6], which are consistent with Young's result [22][25] for this special case. Recently, LeVeque and Trefethen [15] used a tilted grid to interpret Frankel's result in a more intuitive way. The model problem analysis for the SOR method with red/black ordering first appeared in [14] and is included in Chapter 3 of this thesis. For the 9-point stencil case, there have been relatively few results. A rough estimation for the relaxation parameter was given by Garabedian [7]. More recently, a more precise analysis was performed by Adams, LeVeque and Young [1]. In the above two cases, they considered the 9-point SOR method with a single relaxation parameter. The two-level SOR method described in Chapter 4 is a different method which relies on block level as well as point level relaxation iterations and uses two different relaxation parameters. The analysis that was used to study this method is the model problem analysis performed in the transform domain.

7.2 Future Extensions

The Fourier transform domain approach can be rigorously applied, if the differential equation and its discretization satisfy four conditions:

(I) linearity of the differential equation,

(II) constant coefficient functions for the differential operator,

(III) regularity of the problem domain and appropriate boundary conditions,

(IV) a uniform discretization grid.

Due to condition (IV), the Fourier transform domain approach applies to finite-difference and Fourier spectral methods but not to finite-element or Chebyshev spectral methods. In this thesis, except the local relaxation method of Chapter 3, all the analysis was restricted to the model Poisson equation which satisfies all the above conditions. Nevertheless, empirical studies seem to suggest that the range of validity of Fourier analysis, and in particular of local Fourier analysis, is much larger than was previously suspected. It is therefore important to attempt to delineate very precisely the range of validity of Fourier methods. This issue is not merely one of mathematical rigor, but also a practical one, since we would like to know which ones of conditions (I)-(IV) can be relaxed, and which ones cannot. For example, it seems that conditions (II) and (III) are not as necessary as conditions (I) and (IV) in the case of elliptic PDEs [4] (also see Chapter 3). A better understanding of these issues may also lead to modifications and generalizations of current Fourier techniques.

One general future research direction is to examine how digital filtering theory can enrich Fourier methods for solving PDEs. Although digital filtering theory can be viewed as a branch of Fourier analysis, it has developed into an independent and rich field. In this thesis, we showed how to use the Laplace transform and Z-transform to design a mode-dependent finite-difference method in Chapter 2, and the multirate signal processing techniques to study the restriction and interpolation operators in Chapter 5. It is reasonable to believe that more fruitful results may be obtained by investigating further the relationship between digital filtering and numerical PDE algorithms. Another potential research direction is the link between the control theory and the solution of time-dependent PDEs, since stability is the common central theme in these two fields and the generalized Fourier techniques such as the Laplace transform and Z-transform seem to be ideal common analytic tools.

Research on parallel computation is an important trend now as well as in the future. In this thesis, we have considered special purpose computer architectures such as mesh-connected processor arrays and we have examined parallel PDE algorithms such as the local relaxation method. In the future, special-purpose architectures and algorithms for solving hyperbolic and parabolic equations will also need to be developed. Practically speaking, a parallel Navier-Stokes solver would have a much greater impact than the parallel Poisson solver that we have studied here.

7.3 References

[1] L. Adams, R. J. LeVeque, and D. M. Young, "Analysis of the SOR Iteration for the 9-Point Laplacian," To appear in *SIAM J. Num. Analy.*.

[2] D. A. Anderson, J. C. Tannehill, and R. H. Pletcher, *Computational Fluid Mechanics and Heat Transfer*. New York, NY: Hemisphere Publishing Corp., 1984.

[3] A. Brandt , "Multi-level Adaptive Solutions to Boundary-value Problems," *Mathmatics of Computation*, vol. 31, no. 138, pp. 333-390, Apr. 1977.

[4] T. F. Chan and H. C. Elman, "Fourier Analysis of Iterative Methods for Elliptic Problems," Technical Reports 1763, University of Maryland, Department of Computer Science, Jan. 1987.

[5] B. Fornberg, "On a Fourier Method for the Integration of Hyperbolic Equations," *SIAM J. Numer. Analy.*, vol. 12, no. 4, pp. 509-528, Sep. 1975.

[6] S. P. Frankel, "Convergence Rates of Iterative Treatments of Partial Differential Equations," *Math. Tables Aids Comput.*, vol. 4, pp. 65-75, 1950.

[7] P. R. Garabedian, "Estimation of the Relaxation Factor for Small Mesh Size," *Math. Tables Aids Comput.*, vol. 10, pp. 183-185, 1965.

[8] M. Giles and W. T. Thompkins, Jr., "Asymptotic Analysis of Numerical Wave Propagation in Finite Difference Equations," MIT GT&PDL Report 171, Mar. 1983.

[9] D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications*. Philadelphia, PA: SIAM, 1977.

[10] L. A. Hageman and D. M. Young, *Applied Iterative Methods*. New York, N.Y. : Academic Press, Inc. , 1981.

[11] P. Henrici, *Discrete Variable Methods in Ordinary Differential Equations*. New York, N.Y.: John Wiley & Sons, Inc., 1962.

[12] R. Kettler, "Analysis and Comparison of Relaxation Schemes in Robust Multigrid and Preconditioned Conjugate Gradient Methods," in *Multigrid Methods*, ed. W. Hackbusch and U. Trottenberg, New York, N.Y.: Springer-Verlag, 1982, pp. 502-534.

[13] H.-O. Kreiss and J. Oliger, "Comparison of Accurate Methods for the Integration of Hyperbolic Equations," *Tellus*, vol. 24, no. 3, 1972.

[14] C.-C. J. Kuo, B. C. Levy, and B. R. Musicus, "A Local Relaxation Method for Solving Elliptic PDEs on Mesh-Connected Arrays," to appear in *SIAM Journal on Scientific and Statistical Computing*, June 1987.

[15] R. J. LeVeque and L. N. Trefethen, "Fourier Analysis of the SOR Iteration," Numerical Analysis Report 86-6, Department of Mathematics, MIT, Cambridge, MA. ICASE Report 86-93, Sep. 1986.

[16] R. D. Richtmyer and K. W. Morton, *Difference Methods for Initial-Value Problems*. New York, N.Y.: Interscience Publishers, 1967.

[17] H. L. Stone, "Iterative Solution of Implicit Approximations of Multidimensional Partial Differential Equations," *SIAM Journal on Numerical Analysis*, vol. 5, no. 3, pp. 530-558.

[18] K. Stüben and U. Trottenberg, "Multigrid Methods : Fundamental Algorithms, Model Problem Analysis, and Applications," in *Multigrid Methods*, ed. W. Hackbusch and U. Trottenberg, New York, N.Y.: Springer-Verlag, 1982.

[19] L. N. Trefethen, "Group Velocity in Finite Difference Schemes," *SIAM Review*, vol. 24, no. 2, pp. 113-136, Apr. 1982.

[20] L. N. Trefethen, "Dispersion, Dissipation, and Stability," in *Proceedings of the 1985 Dundee Conference in Numerical Analysis*, Oxford U. Press, 1986.

[21] L. N. Trefethen, "Stability of Finite-Difference Models Containing Two Boundaries or Interfaces," *Math. Comp.*, vol. 45, no. 172, 1985.

[22] R. S. Varga, *Matrix Iterative Analysis*. Englewood Cliffs, N.J. : Prentice-Hall, Inc. , 1962.

[23] R. Vichnevetsky, *Fourier Analysis of Numerical Approximations of Hyperbolic Equations*. Philadelphia, PA: SIAM, 1982.

[24] R. G. Voigt, D. Gottlieb, and M. Y. Hussaini, *Spectral Methods for Partial Differential Equations*. Philadephia: SIAM, 1984.

[25] D. M. Young , "Iterative Methods for Solving Partial Differential Equations of Elliptic Type ," Doctoral Thesis , Harvard University , 1950.