MGCG METHOD:
A ROBUST AND HIGHLY PARALLEL ITERATIVE METHOD
MGCG K!: %m%P%9%H$G9b8zN($JJBNs2rK!

by

Osamu Tatebe

7zIt =$8+

A Thesis

Submitted to

The Graduate School of
The University of Tokyo
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Science
in Information Science

December 1996

ABSTRACT

A multigrid preconditioned conjugated gradient (MGCG) method, which is an iterative method to solve a large sparse matrix, is proposed. The MGCG method is a conjugate gradient method preconditioned by the multigrid method. First, sufficient conditions for a valid multigrid preconditioner are given. Next, the rate of convergence of the MGCG method is investigated for the Poisson equation with constant diffusion coefficient. Local modes of Fourier analysis are useful to analyze all eigenvalues of the multigrid preconditioned matrix, which give an upper bound of the rate of convergence by Chebyshev polynomials. Consequently, the rate of convergence is quite good and independent of the problem size for the two-dimensional Poisson equation. For strongly discontinuous coefficients, the efficient convergence property and robustness of the MGCG method are shown by numerical eigenvalue analysis. This thesis, moreover, studies the parallel MGCG method and its efficient implementation on distributed memory machines. Though there is a trade-off between parallel efficiency and the rate of convergence, it is shown that the parallel MGCG method has both much better rate of convergence and higher parallel efficiency than other methods by evaluating it on stock multicomputers.

O@ J8 MW ;]

Bg5,LOAB9TNs$NH?I|2rK!$G$"$k%^%k%A%0%j%C%IA0=hM}IU$-6&Lr8{G[K! (MGCG K!) $NDs0F$r9T$&.
MGCG K!$O%^%k%A%0%j%C%IK! (MG K!) Kh$jA0=hM}$r9T$&6&Lr8{G[K!$G$"$k. $^$: MG K!$,6&Lr
8{G[K!$NA0=hM}$H7F$N>r7o$rK~$?$9$?$a$N==J,>r7o$rM?$($k. <!$K, 3H;678?t$,0lDj$N%]%"%=%sJ}
<0KBP7$F MG A0=hM}8e$N9TNs$N8GM-CM2r@O$r9T$$, MGCG K!$N<}B+N($N2r@O$r9T$&. $=$l$>$l$N%0
%j%C%IK*1k%U!<%j%(2r@O$N%m!<%+%k%b!<%I$rMQ$$$k3HKh$j, A0=hM}8e$N9TNs$NA4$F$N8GM-CM$r
@OE*$K5a$ak3HK@.8y$7, %A%'%S%7%'%UB?9'<0$rMQ$$$F MGCG K!$NJ?6Q<}B+N(, A26a<}B+N($N>e8B$
5aa?. 3N7k2L, 2 <!85$N%]%"%=%sJ}Dx<0$G$O MGCG K!$N<}B+N($O6K$aFNI/, 7+$bLdBj%5%$%:$K
hi$J$$$3H,J,$+$C$?. $^$?HsO"B3$J3H;678?t$r;}$D%]%"%=%sJ}Dx<0$KBP7F$O?tCME*$K8GM-CM2r
$r9T$&3HKh$j, MGCG K!$O$=$liNLdBjKBP7FbNI$$<}B+@-$r;}$A, %m%P%9%H$J2rK!$G$"$k3
$,J,$+C?. 5i$K MGCG K!$NJBNs2=, J,;6%a%b%j7?JBNs7W;;5!>eXN8zN(E*$J<BAu$N8&5f$r9T$$, JI
Ns8zN($H<}B+@-$O8_$$$K%H%l!<%I%*%U$N4X78$K$"$k$,, <B5!$GJBNs MGCG K!$NI>2A$r9T$&$3HKhj
%H%l!<%I%*%U$NE@$G$OB>$N2rK!KHfY, 9b$$JBNs8zN($HNI$$<}B+N($r9g$o$;;}$DM%l?2rK!HJk3$H
J,$+$C$?.

Acknowledgments

I would like to give my great thanks for my supervisor Prof. Y. Oyanagi. He has introduced me into quite a hot area of high performance computing and parallel numerical methods, and led to this exciting work.

I am grateful to Prof. K. Murata at Kanagawa University. He gave me quite good comments and advised me to investigate the MGCG method on the Poisson problem with severe coefficient jumps. I would like to thank Prof. M. Mori and Prof. M. Sugihara at the University of Tokyo. They suggested me to analyze eigenvalue distribution of the multigrid preconditioned matrix, which gave a definite evaluation of the MGCG method.

I would like to thank Prof. T. Nodera at Keio University for valuable comments about Lanczos method and CG methods. I would like to thank Prof. S. L. Zhang at Tsukuba University for valuable discussion about Lanczos polynomial and three-term recurrence. He also sent me historical papers of Lanczos and Stiefel.

I appreciate discussions with Dr. R. Suda and members of Oyanagi laboratory. Dr. R. Suda gave me appropriate comments about numerical methods.

I would like to thank the members of Fujitsu Parallel Computing Research Facilities for providing and supporting AP1000 multicomputer. In particular, I am grateful to Dr. T. Horie and Dr. K. Hayashi. They have settled my questions about AP1000 and explained details of an implementation of its message handling.

Table of Contents

List of Tables

List of Figures

Chapter 1

INTRODUCTION

1.1 Overview

The advent of parallel machines in the last decade has brought huge computational power and provided possibilities of genuine large-scale numerical simulations. However, because of communication overhead and load imbalance it is difficult to gain its peak performance, thus it is quite important to develop parallel algorithms that can be implemented efficiently on distributed memory machines.

In general, there is a trade-off between high parallelism and efficient algorithm; for example, the Jacobi method for the solution of a large sparse matrix has quite high parallelism, however it has poor convergence factor, while the ICCG method or robust multigrid method converges efficiently but has poor parallelism, thus it is desirable that new algorithm has adequate parallelism and suitable efficiency.

1.2 Iterative Methods for Large Sparse Matrix

Most computation of numerous large-scale simulations is spent in the solution of a large sparse matrix that arises after discretization of partial differential equations. Typical numerical methods for the sparse matrix are the *conjugate gradient* (CG) method [26] and the *multigrid* (MG) method [15]. The CG method came into widespread use from the early 1970s because of its optimality over the solution space, fast convergence using preconditioners, small memory size and acceptable rounding-off error properties. Recently, generalization of the CG method for nonsymmetric and/or indefinite matrix with these remarkable properties is a hot research area [48, 55, 19, 63]. The MG method was realized to be quite an efficient method with a broad area of application from the mid-seventies because of its mesh-independent and fast convergence, and the number of publications

has grown rapidly.

This thesis proposes the *multigrid preconditioned conjugate gradient* (MGCG) method that is a CG method preconditioned by the MG method, and shows a sufficient condition of the MG preconditioner. Several numerical examples show that this method is quite efficient for the Poisson equation with uniform or strongly discontinuous diffusion coefficient [52]. Recently, many researchers use and study the MGCG method, and numerical results of actual applications are obtained. For stretched grids and convection diffusion problems, it is reported the MGCG method with an alternating line Gauss-Seidel smoother is robust [58], and for numerical simulations of groundwater flow, it is also reported that the MGCG method with semicoarsening is quite efficient [3].

This thesis investigates the rate of convergence of the MGCG method for the Poisson equation. There are several studies in the rate of convergence of the MG method by estimating the minimal eigenvalue of the iteration matrix [21, 8, 59]. Moreover, to predict a practical convergence factor, a smoothing factor that is a contraction number of the smoothing method with respect to rough components has been researched [59, 61]. For a sharp estimation of the rate of convergence of the MGCG method, the distribution of eigenvalues of the preconditioned matrix is necessary, since the rate of convergence of the CG method strongly depends on the distribution [7]. This thesis studies the distribution of eigenvalues of the MG preconditioned matrix using Fourier components on each grid level, which are quite a useful tool for obtaining the smoothing factor. Exploiting Fourier components on each grid level, all the eigenvalues of the MG preconditioned matrix can be analytically obtained. This thesis investigates the eigenvalues of the two-grid preconditioned matrix with the damped Jacobi smoother or the Red-Black Gauss-Seidel (RB-GS) smoother for 1- and 2-dimensional Poisson equations. Consequently, in one dimension, we show all the eigenvalues of the MG preconditioned matrix with RB-GS smoother lie at unity, that is, the MG preconditioner is an exact solver. In two dimensions, the eigenvalues of the two-grid preconditioned matrix exist between $\frac{3}{4}$ and 1 and its asymptotic rate of convergence is 1.14 independently of the mesh size.

This thesis also investigates the rate of convergence for the Poisson equation with severe coefficient jumps. For such problem, no theoretical result has been obtained, however, there is a special case whose MG preconditioned matrix has the same spectrum, i. e. the same eigenvalue distribution, as that for the Poisson equation with constant diffusion coefficient. First, the special case is considered and its proof is given, and then more complex problems are numerically studied. From the numerical results, the spectrum of the MG preconditioned matrix is nearly the same as that for uniform diffusion constant problem, unless the problem cannot be discretized or approximated correctly on the coarsest grid. Otherwise, the MG preconditioned matrix has a few isolated

eigenvalues, and the number of isolated eigenvalues is expected to depend on the mesh size of the coarsest grid that can approximate the problem correctly.

1.3 Parallelization of Iterative Methods

For fast convergence, the CG method generally uses a ILU preconditioner. However, the ILU preconditioner has poor parallelism and it has been reported that the ILU preconditioned CG method is slower than the SCG method, which has high parallelism and cheap rate of convergence, on vector machines [23].

The MG method converges quite fast and independently of the mesh size, however, for robustness it should use a strong smoother; line relaxation, plain relaxation or ILU [1, 11]. Since the strong smoother has poor parallelism, the MG method does not gain high parallel efficiency. In fact, Hempel [25] reported that a parallel efficiency of the multigrid method with the robust smoother gains only about 60%.

This thesis studies a parallelization of the MGCG method and efficient implementation on distributed memory machines [53]. Investigating the rate of convergence of the MGCG method, the MGCG method with highly parallel relaxation; RB-GS, four color GS and so on, has quite a good rate of convergence and mesh-independent convergence property, therefore, basically the MGCG method has ample data-parallelism, however on quite coarse grids, the network latency is not hidden by overlapping computation and communication, and the ratio of communication becomes high, so the communication overhead is critical. Moreover, when the number of data-parallelism is smaller than the number of processors, there are idle processors, thus, in order that the parallel MGCG method achieves high performance, the optimal MG schedule is considered a trade-off between the convergence rate and parallel efficiency of a target architecture.

1.4 Organization

This thesis is organized as follows. Chapter 2 briefly describes mathematical preliminary including vector/matrix norm, matrix properties; positive definite matrix, M-matrix and Stieltjes matrix, and Krylov subspace.

In Chapter 3, basic iterative methods and their convergence theorems are explained. Basic iterative methods are usually used as smoothers of MG methods. This chapter also explains the rate of convergence and convergent splitting.

The CG method and the preconditioned CG (PCG) method are explained in Chapter 4. This chapter intends to explain the CG method intuitively relating its optimality over the Krylov sub-

space. The rate of convergence of the CG method is also explained.

Chapter 5 explains the MG method. In this chapter, the mesh-independent convergence property and a smoothing factor that shows a practical convergence factor are also explained.

In Chapter 6, the MGCG method is formulated. Sufficient conditions for two-grid preconditioners are given in Section 6.1 and they are extended to the MG preconditioner.

In Chapter 7, exploiting Fourier components on each grid level, all the eigenvalues of the MG preconditioned matrix are analytically obtained, and it is proved that the MGCG method is robust and has quite a fast rate of convergence.

Chapter 8 investigates the rate of convergence of the MGCG method for the Poisson equation with severe coefficient jumps. First, a special case whose MG preconditioned matrix has the same spectrum as that for the Poisson equation with constant diffusion coefficient is considered, and then more complex case is numerically analyzed by eigenvalue analysis.

Chapter 9 studies a parallelization of the MGCG method and its efficient implementation on distributed memory machines. On quite coarse grids, parallel efficiency is low due to the communication overhead and load imbalance, while the computation on these coarse grids is necessary to have the mesh-independent convergence property, thus the optimal MG schedule depends on a target architecture. In this chapter, it is evaluated on Fujitsu multicomputer AP1000 and AP1000+, and the parallel efficiency is also evaluated.

The main conclusions and future research are summarized in Chapter 10.

Chapter 2

MATHEMATICAL PRELIMINARY

2.1 Norms

Definition 2.1.1 (Norm) $\|\cdot\|$ is called *norm* if and only if

1. $\|\boldsymbol{x}\| > 0$ for $\forall \boldsymbol{x} \neq 0$ and $\|\boldsymbol{x}\| = 0$ for $\boldsymbol{x} = 0$
2. $\|\alpha \boldsymbol{x}\| = \alpha \|\boldsymbol{x}\|$
3. $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$

Definition 2.1.2 $\|\boldsymbol{x}\| = (\boldsymbol{x}^H \boldsymbol{x})^{\frac{1}{2}}$ is called the *Euclidean norm.*

Definition 2.1.3 (Spectral radius) Let A be an $n \times n$ complex matrix,

$$\rho(A) = \max\{|\lambda| \mid \lambda \in S(A)\}, \tag{2.1}$$

is called the *spectral radius* of A, where

$$S(A) = \{\lambda \mid \lambda \text{ is an eigenvalue of } A\}, \tag{2.2}$$

is the *spectrum* of A.

Definition 2.1.4 (Spectral norm) Let A be a $n \times n$ complex matrix,

$$\|A\| \equiv \sup_{\boldsymbol{x} \neq 0} \frac{\|A\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \tag{2.3}$$

is called the *spectral norm* of A, where $\|\cdot\|$ is the Euclidean norm.

Corollary 2.1.1 *For any $n \times n$ complex matrix,*

$$\|A\| \geq \rho(A). \tag{2.4}$$

If A is normal, i. e. $A^H A = AA^H$, or hermitian,

$$\|A\| = \rho(A). \tag{2.5}$$

Proof. See Varga [56]. □

2.2 Matrix Properties

Definition 2.2.1 (Positive Definite) A real matrix A is called *positive(semi-positive or non-negative/negative/semi-negative or non-positive) definite* if $\boldsymbol{x}^T A\boldsymbol{x} > 0 (\geq 0 / < 0 / \leq 0)$ for $\forall \boldsymbol{x} \neq 0$.

The order relation used in the following definition is the usual component-wise order; with $A = [a_{ij}]$ and $B = [b_{ij}]$, then $A \leq B$ if $a_{ij} \leq b_{ij}$.

Definition 2.2.2 The real $n \times n$ matrix $A = [a_{ij}]$ is

1. *Diagonally dominant* if

$$|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|, \quad i = 1,\ 2,\ \ldots,\ n \tag{2.6}$$

   and *strictly diagonally dominant* if strict inequality holds in Eq. (2.6) for all i.

2. *Reducible* if there is a permutation matrix P such that

$$PAP^T = \begin{bmatrix} B_{11} & B_{12} \\ 0 & B_{22} \end{bmatrix} \tag{2.7}$$

   and *irreducible* if it is not reducible.

3. *Irreducibly diagonally dominant* if it is diagonally dominant, irreducible, and strict inequality holds in Eq. (2.6) for at least one i.

Definition 2.2.3 A real matrix A is called *monotone* if $A\boldsymbol{x} \geq 0$ implies $\boldsymbol{x} \geq 0$.

Definition 2.2.4 (M-matrix) A real square matrix $A = [a_{ij}]$ is called a (nonsingular) *M-matrix* if $a_{ij} \leq 0$ for $i \neq j$ and if it is monotone, that is, if $A^{-1} \geq 0$.

Definition 2.2.5 (Stieltjes matrix) Let a real square matrix $A = [a_{ij}]$ be positive definite. If $a_{ij} \leq 0$ for $i \neq j$, the matrix is called the *Stieltjes matrix*.

2.3 Krylov Subspace

Definition 2.3.1 (Krylov subspace) $K_n(A, \boldsymbol{r}) = \text{Span}\{\boldsymbol{r},\ A\boldsymbol{r},\ \ldots,\ A^{n-1}\boldsymbol{r}\}$ is called the *Krylov subspace* associated with A and $\boldsymbol{r}$ [14].

Chapter 3

BASIC ITERATIVE METHODS

This chapter explains basic iterative methods for solving linear systems. Basic iterative methods are popularly used for smoothers of multigrid methods. There are many good references; for examples, Varga [56] and Young [62].

3.1 Basic Iterative Methods

Consider a system of linear equations

$$A\boldsymbol{x} = \boldsymbol{b}, \tag{3.1}$$

where A is a n by n nonsingular matrix. Basic iterative methods split the matrix A into $P - Q$ and update the kth approximate solution by

$$P\boldsymbol{x}_k = Q\boldsymbol{x}_{k-1} + \boldsymbol{b} \tag{3.2}$$

with $\boldsymbol{x}_0$ as an initial approximation. P is a nonsingular matrix sometimes called the *preconditioning* matrix, and Q is called the *defect* matrix. $A = P - Q$ is called a *splitting* of A. Note that if $P = A$, then $Q = 0$, and the exact solution $\hat{\boldsymbol{x}} = A^{-1}\boldsymbol{b}$ is obtained after one step of this method. Since the iteration of Eq. (3.2) needs a linear solution with P at each step, P is chosen such that the linear equation can be solved with reasonable cost, while P should be chosen nearly A for a faster rate of convergence that is described later.

From Eq. (3.2), the kth approximation $\boldsymbol{x}_k$ is written using the initial approximation $\boldsymbol{x}_0$:

$$\boldsymbol{x}_k = (P^{-1}Q)^k\boldsymbol{x}_0 + \sum_{i=0}^{k-1}(P^{-1}Q)^i P^{-1}\boldsymbol{b}. \tag{3.3}$$

If we let the initial approximation be the exact solution, all approximations are equal to the exact solution from Eq. (3.2), which means the exact solution is a fixed point of Eq. (3.2). Then Eq. (3.2) is said to be *consistent* with Eq. (3.1).

3.2 Convergence Factor and Rate of Convergence

To study convergence condition of iterative methods, the following lemma is necessary.

Lemma 3.2.1 *For an arbitrary square matrix A,*

1. $\lim_{k\to\infty} A^k = 0 \Longleftrightarrow \rho(A) < 1$.
2. *If* $\rho(A) < 1$, $(I - A)^{-1} = (I + A + A^2 + \cdots)$ *is convergent.*

Let the error $\boldsymbol{e}_k = \hat{\boldsymbol{x}} - \boldsymbol{x}_k$ of the kth iteration, where $\hat{\boldsymbol{x}}$ is the solution. Eq. (3.2) gives

$$\boldsymbol{e}_{k+1} = P^{-1}Q\boldsymbol{e}_k, \tag{3.4}$$

hence

$$\|\boldsymbol{e}_{k+1}\| = \|P^{-1}Q\boldsymbol{e}_k\| \le \|P^{-1}Q\|\ \|\boldsymbol{e}_k\|. \tag{3.5}$$

$P^{-1}Q$ is called the *iteration matrix*, and $\|P^{-1}Q\|$ is called the *contraction number*.

Definition 3.2.1 (The convergence factor) $\frac{\|\boldsymbol{e}_k\|}{\|\boldsymbol{e}_0\|}$ is called the *convergence factor* for k steps, and $R_k = \left(\frac{\|\boldsymbol{e}_k\|}{\|\boldsymbol{e}_0\|}\right)^{\frac{1}{k}}$ is called the *average convergence factor*. $R_\infty = \lim_{k\to\infty} R_k$ is called the *asymptotic convergence factor*.

Definition 3.2.2 (The rate of convergence) $r_k = -\log_{10} R_k$ is called the *average rate of convergence*, and $r = -\lim_{k\to\infty} \log_{10} R_k$ is called the *(asymptotic) rate of convergence*.

Note that r_k and r are estimates of the number of new correct decimals per iteration. From Eq. (3.4),

$$\|\boldsymbol{e}_k\| = \|(P^{-1}Q)^k\boldsymbol{e}_0\| \le \|(P^{-1}Q)^k\|\ \|\boldsymbol{e}_0\|. \tag{3.6}$$

Thus the convergence factor of the iterative method can be estimated by $\|(P^{-1}Q)^k\|$. By Lemma 3.2.1, this method converges if and only if $\rho(P^{-1}Q) < 1$. For the convergence factor, the following theorem holds.

Theorem 3.2.2 *Let H be an $n \times n$ complex matrix,*

$$\lim_{k\to\infty} \|H^k\|^{\frac{1}{k}} = \rho(H), \tag{3.7}$$

where $\|\cdot\|$ is a matrix norm.

Proof. See Varga [56]. □

From Theorem 3.2.2, the asymptotic convergence factor is obtained by $\rho(P^{-1}Q)$.

3.3 Convergent Splitting

Eq. (3.2) converges if and only if the splitting is a convergent splitting.

Definition 3.3.1 (Convergent splitting) If $\rho(P^{-1}Q) < 1$, this splitting is called the *convergent splitting*.

First, we consider a symmetric and positive definite matrix. The following splitting is useful.

Definition 3.3.2 (P-regular splitting) If P is nonsingular and (the symmetric part of) $P+Q$ is positive definite, this splitting is called the *P-regular splitting*, where the symmetric part of C is $(C+C^T)/2$.

To prove that a P-regular splitting for a symmetric and positive definite matrix is convergent, the Stein's theorem [49] is necessary. For a proof, see, for example, Ortega [41].

Theorem 3.3.1 (Stein 1952) *If B is an $n \times n$ complex matrix, then $\rho(B) < 1$ if and only if there is a hermitian positive definite matrix A such that $A - B^H AB$ is positive definite.*

Theorem 3.3.2 (P-regular splitting theorem) *If A is symmetric and positive definite, and if $A = P - Q$ is a P-regular splitting, then $\rho(P^{-1}Q) < 1$.*

Proof. Let $H = P^{-1}Q$ and $C = A - H^T AH$. Then, since $P^{-1}Q = I - P^{-1}A$, we have

$$\begin{aligned} C &= A - (I - P^{-1}A)^T A(I - P^{-1}A) \\ &= (P^{-1}A)^T (P^T + Q)(P^{-1}A). \end{aligned} \tag{3.8}$$

Since $P+Q$ is positive definite, so that P^T+Q is also positive definite. Thus, since C is congruent to P^T+Q, it is positive definite. Because A is positive definite, the proof is complete using Stein's Theorem 3.3.1 □

From Theorem 3.3.2, P-regular splitting is also a convergent splitting if A is symmetric. There are two different converses associated with Theorem 3.3.2. These proofs are given by, for example, Ortega [42].

Theorem 3.3.3 (First converse of P-regular splitting) *Assume that A is symmetric and nonsingular, $A = P - Q$ is a P-regular splitting, and $\rho(P^{-1}Q) < 1$. Then A is positive definite.*

Theorem 3.3.4 (Second converse of P-regular splitting) *Assume that $A = P - Q$ is symmetric and nonsingular, P is symmetric positive definite, and $\rho(P^{-1}Q) < 1$. Then A and $P+Q$ are positive definite.*

When A is a diagonally dominant matrix or an M-matrix, the following splitting is considered.

Definition 3.3.3 (Weak regular splitting and regular splitting) $A = P - Q$ is a *weak regular splitting* [43] if P is nonsingular, $P^{-1} \geq 0$ and $P^{-1}Q \geq 0$. It is a *regular splitting* [56] if $P^{-1} \geq 0$ and $Q \geq 0$.

Clearly a regular splitting is a weak regular splitting. The following convergence theorem holds. For a proof, see [42] or [7].

Theorem 3.3.5 (Weak regular splitting theorem) *If $A = P - Q$ is a weak regular splitting, then the splitting is convergent if and only if A is monotone.*

3.4 Richardson Method

The Richardson method is quite a simple method. Let $A = I - Q$, where I is the identity matrix, the *Richardson method* iterates

$$\begin{aligned} \boldsymbol{x}_{k+1} &= (I - A)\boldsymbol{x}_k + \boldsymbol{b} \\ &= \boldsymbol{x}_k + \boldsymbol{r}_k, \end{aligned} \tag{3.9}$$

where $\boldsymbol{r}_k = \boldsymbol{b} - A\boldsymbol{x}_k$. Thus, the residual $\boldsymbol{r}$ is added to the approximation at each iteration. From Eq. (3.9),

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_0 + \sum_{l=0}^{k} \boldsymbol{r}_l \tag{3.10}$$

Without loss of generality, we assume $\boldsymbol{x}_0 = 0$, since if we let $\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{x}_0$ for any initial vector $\boldsymbol{x}_0$, the iterative method is equal to that with zero initial vector $\boldsymbol{y}_0$ for the linear transformed linear equation $A\boldsymbol{y} = \boldsymbol{b} - A\boldsymbol{x}_0 = \tilde{\boldsymbol{b}}$. From Eq. (3.10),

$$\boldsymbol{x}_{k+1} = \sum_{l=0}^{k} \boldsymbol{r}_l = \sum_{l=0}^{k} (I - A)^l \boldsymbol{r}_0 = Q_k(A)\boldsymbol{r}_0 \in K_{k+1}(A, \boldsymbol{r}_0), \tag{3.11}$$

where $Q_k(A)$ is a polynomial of degree k. $K_{k+1}(A, \boldsymbol{r}_0)$ is the *Krylov subspace* which is the space spanned by $\boldsymbol{r}_0,\ A\boldsymbol{r}_0,\ \ldots,\ A^k\boldsymbol{r}_0$. Since $\boldsymbol{b} = \boldsymbol{r}_0$,

$$\begin{aligned} \boldsymbol{r}_{k+1} &= \boldsymbol{b} - AQ_k(A)\boldsymbol{r}_0 \\ &= P_{k+1}(A)\boldsymbol{r}_0, \end{aligned}$$

where $P_{k+1}(A)$ is a polynomial of degree $(k+1)$. Note that $P_{k+1}(0) = 1$. The polynomial $P_{k+1}(A) = (I - A)^{k+1}$ is called the *residual polynomial*, which characterizes the Richardson iteration.

Theorem 3.4.1 *The Richardson method converges if and only if* $\rho(I - A) < 1$.

Proof. The iteration matrix of the Richardson method is $I - A$, thus the theorem holds. □

If A has real eigenvalues, the Richardson iterations converge if and only if $0 < \lambda < 2$, where λ is an eigenvalue of A. The asymptotic convergence factor is

$$\max(|1 - \lambda_{\max}|,\ |1 - \lambda_{\min}|). \tag{3.12}$$

The Richardson method is modified by relaxation parameter ω, and this method is called the *damped Richardson method* in this thesis. In this case, the splitting is $A = \frac{1}{\omega}I - (\frac{1}{\omega}I - A)$. The residual polynomial and convergence condition can be similarly analyzed. The residual polynomial is $P_{k+1}(A) = (I - \omega A)^{k+1}$. The damped Richardson method is convergent if and only if $\rho(I - \omega A) < 1$. The asymptotic convergence factor is $\max(|1 - \omega\lambda_{\max}|,\ |1 - \omega\lambda_{\min}|)$.

3.5 Damped Jacobi Method

Let $A = D - Q$, where D is the diagonal part of A, the *Jacobi method* for $A\boldsymbol{x} = \boldsymbol{b}$ iterates

$$D\boldsymbol{x}_k = Q\boldsymbol{x}_{k-1} + \boldsymbol{b}, \tag{3.13}$$

with $\boldsymbol{x}_0$ an initial approximation. The following two convergence theorems hold.

Theorem 3.5.1 *If* $A = D - Q$ *is symmetric and positive definite, then the Jacobi iterations converge for any* $\boldsymbol{x}_0$ *if and only if* $D + Q$ *is positive definite.*

Proof. If $D + Q$ is positive definite, then $A = D - Q$ is a P-regular splitting, thus the Jacobi method converges. Conversely, if the Jacobi method converges, $D + Q$ is positive definite, since A and D are symmetric and positive definite. □

Theorem 3.5.2 *If* A *is strictly or irreducibly diagonally dominant, or if* A *is an* M*-matrix, then the Jacobi iterations converge for any* $\boldsymbol{x}_0$.

Proof. $A = D - Q$ is clearly a regular splitting, thus the Jacobi iterations converge by Theorem 3.3.5. □

Using a relaxation parameter ω, the *damped Jacobi method* for $A\boldsymbol{x} = \boldsymbol{b}$ iterates

$$\frac{1}{\omega}D\boldsymbol{x}_k = \Big(\frac{1}{\omega}D - A\Big)\boldsymbol{x}_{k-1} + \boldsymbol{b} \tag{3.14}$$

or

$$D\boldsymbol{x}_k = ((1 - \omega)D + \omega Q)\boldsymbol{x}_{k-1} + \omega\boldsymbol{b}, \tag{3.15}$$

where D is the diagonal part of A and $Q = D - A$. The following theorem holds.

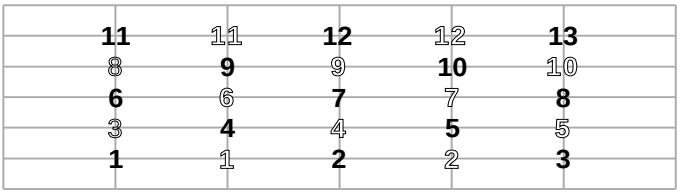

Figure 3.1: Red-Black ordering

Theorem 3.5.3 *If the Jacobi method converges, the damped Jacobi method converges if and only if* $0 < \omega \leq 1$.

Damping is introduced to satisfy the smoothing property described in Chapter 5, thus the damped Jacobi method is only used as a smoother of the multigrid method.

3.6 Red-Black Gauss-Seidel Method

Let $A = D - L - U$, where D is diagonal, L is strictly lower triangular and U is strictly upper triangular matrix, the *Gauss-Seidel* method for $A\boldsymbol{x} = \boldsymbol{b}$ iterates

$$(D - L)\boldsymbol{x}_k = U\boldsymbol{x}_{k-1} + \boldsymbol{b} \tag{3.16}$$

with $\boldsymbol{x}_0$ an initial approximation. The convergence theorem of the Gauss-Seidel method has been given.

Theorem 3.6.1 *If A is strictly or irreducibly diagonally dominant, or if A is an M-matrix, then the Gauss-Seidel iterations converge for any $\boldsymbol{x}_0$ to the unique solution.*

The Gauss-Seidel method requires solving the lower triangular matrix $(D - L)$ in each iteration, thus it has poor parallelism. However, when A has property A, the Gauss-Seidel method can be parallelized by reordering.

Definition 3.6.1 (Property A) (Young 1950) If there is a permutation matrix P such that

$$PAP^T = \begin{bmatrix} D_1 & U \\ L & D_2 \end{bmatrix} \tag{3.17}$$

where D_1 and D_2 are diagonal matrices, A is said to have *property A*.

A typical example of matrices with property A occurs for second-order elliptic differential equations when we use the standard five-point difference approximation. The canonical form for the matrix

is obtained when we use a so-called red-black (or white-black as on a chessboard) ordering of the mesh points depicted by Figure 3.1. The Gauss-Seidel method with red-black ordering is called the *Red-Black Gauss-Seidel* (RB-GS) method.

Chapter 4

CONJUGATE GRADIENT METHOD

M. R. Hestenes and E. Stiefel [26] proposed the conjugate gradient (CG) method in 1952. The CG method is usually introduced by minimizing the quadratic function using conjugate directions, however this chapter intends to explain the CG method intuitively relating its optimality over the Krylov subspace.

4.1 Conjugate Gradient Method

Consider a system of linear equations:

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

where A is an $N \times N$ symmetric and positive definite (s.p.d.) matrix. The CG method minimizes the quadratic function

$$Q(\boldsymbol{x}_i) = \frac{1}{2}\boldsymbol{x}_i^T A\boldsymbol{x}_i - \boldsymbol{b}^T\boldsymbol{x}_i \tag{4.2}$$

over $\boldsymbol{x}_0 + K_i(A, \boldsymbol{r}_0)$ at the ith iteration to solve the linear equation, where $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$ is the residual for an initial approximation $\boldsymbol{x}_0$. Without loss of generality we assume $\boldsymbol{x}_0 = 0$, so that the CG method minimizes $Q(\boldsymbol{x})$ over Krylov subspace $K_i(A, \boldsymbol{r}_0)$. Equality of minimization of $Q(\boldsymbol{x})$ and the solution of the linear equation can be readily shown as follows. Because A is s.p.d., the following relations hold:

$$\|\boldsymbol{y}\|_A^2 = (\boldsymbol{y}, \boldsymbol{y})_A = (\boldsymbol{y}, A\boldsymbol{y}) \geq 0, \text{ equality holds iff } \boldsymbol{y} = 0 \tag{4.3}$$

and

$$(\boldsymbol{y}, \boldsymbol{z})_A = (\boldsymbol{z}, \boldsymbol{y})_A. \tag{4.4}$$

From Eq. (4.3) and (4.4), Eq. (4.2) is rewritten as

$$Q(\boldsymbol{x}_i) = \frac{1}{2}\|\hat{\boldsymbol{x}} - \boldsymbol{x}_i\|_A^2 + C \geq C, \tag{4.5}$$

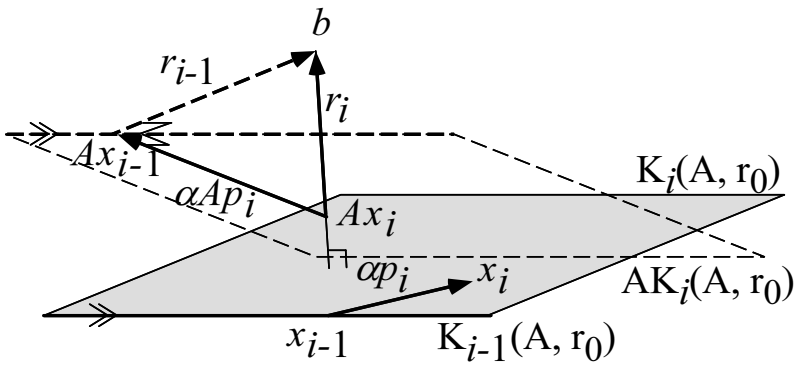

Figure 4.1: Conjugate direction

where $\hat{\boldsymbol{x}}$ is the solution and $C = \frac{1}{2}\boldsymbol{b}^T\hat{\boldsymbol{x}}$ is constant, thus $Q(\boldsymbol{x})$ has a minimum value at $\boldsymbol{x} = \hat{\boldsymbol{x}}$. From Eq. (4.5), it can be shown the error of the ith iteration has the A-orthogonal (conjugate) condition:

$$\hat{\boldsymbol{x}} - \boldsymbol{x}_i \perp_A K_i(A, \boldsymbol{r}_0), \tag{4.6}$$

which means $(\hat{\boldsymbol{x}} - \boldsymbol{x}_i, \boldsymbol{y})_A = 0$ for $\forall \boldsymbol{y} \in K_i(A, \boldsymbol{r}_0)$. This is understandable as follows. Let $\hat{\boldsymbol{x}} - \boldsymbol{x}_i \perp_A K_i(A, \boldsymbol{r}_0)$ and $\forall \boldsymbol{y} \in K_i(A, \boldsymbol{r}_0)$,

$$\|\hat{\boldsymbol{x}} - \boldsymbol{y}\|_A^2 = \|(\hat{\boldsymbol{x}} - \boldsymbol{x}_i) - (\boldsymbol{y} - \boldsymbol{x}_i)\|_A^2 = \|\hat{\boldsymbol{x}} - \boldsymbol{x}_i\|_A^2 + \|\boldsymbol{y} - \boldsymbol{x}_i\|_A^2 \geq \|\hat{\boldsymbol{x}} - \boldsymbol{x}_i\|_A^2, \tag{4.7}$$

since $\hat{\boldsymbol{x}} - \boldsymbol{x}_i \perp_A \boldsymbol{y} - \boldsymbol{x}_i$, thus $Q(\boldsymbol{x})$ has a minimum value at $\boldsymbol{y} = \boldsymbol{x}_i$. Eq. (4.6) follows the following orthogonal condition with respect to $\boldsymbol{r}$

$$\boldsymbol{r}_i \perp K_i(A, \boldsymbol{r}_0), \tag{4.8}$$

because

$$(\hat{\boldsymbol{x}} - \boldsymbol{x}_i, \boldsymbol{y})_A = (A(\hat{\boldsymbol{x}} - \boldsymbol{x}_i), \boldsymbol{y}) = (\boldsymbol{r}_i, \boldsymbol{y}) = 0 \tag{4.9}$$

for $\forall \boldsymbol{y} \in K_i(A, \boldsymbol{r}_0)$. Since $\boldsymbol{x}_i \in \boldsymbol{x}_0 + K_i(A, \boldsymbol{r}_0)$,

$$\boldsymbol{r}_i \in K_{i+1}(A, \boldsymbol{r}_0), \tag{4.10}$$

hence

$$\boldsymbol{r}_{i+1} - \boldsymbol{r}_i \perp K_i(A, \boldsymbol{r}_0), \tag{4.11}$$

which is illustrated by Fig. 4.1. Note that $\{\boldsymbol{r}_0, \boldsymbol{r}_1, \ldots, \boldsymbol{r}_i\}$ is an orthogonal base of Krylov subspace $K_{i+1}(A, \boldsymbol{r}_0)$. Let $-\alpha_i A\boldsymbol{p}_i = \boldsymbol{r}_{i+1} - \boldsymbol{r}_i$, $\boldsymbol{r}_{i+1}$ is updated using $A\boldsymbol{p}_i$ such that $A\boldsymbol{p}_i \in AK_{i+1}(A, \boldsymbol{r}_0)$ and $A\boldsymbol{p}_i \perp K_i(A, \boldsymbol{r}_0)$.

$$\boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i A\boldsymbol{p}_i. \tag{4.12}$$

This is also means

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i \tag{4.13}$$

such that

$$\boldsymbol{p}_i \in K_{i+1}(A, \boldsymbol{r}_0) \quad \text{and} \quad \boldsymbol{p}_i \perp_A K_i(A, \boldsymbol{r}_0), \tag{4.14}$$

thus the $(i+1)$th approximate $\boldsymbol{x}_{i+1}$ is updated by the previous approximate $\boldsymbol{x}_i$ and $\boldsymbol{p}_i$. If $\boldsymbol{p}_i$ is given, α_i is determined by minimization of $Q(\boldsymbol{x})$. From Eq. (4.2) and (4.13),

$$\begin{aligned} Q(\boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i) &= \frac{1}{2}(\boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i)^T A(\boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i) - \boldsymbol{b}^T(\boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i) \\ &= \frac{1}{2}\boldsymbol{p}_i^T A \boldsymbol{p}_i \alpha_i^2 - \boldsymbol{p}_i^T \boldsymbol{r}_i \alpha_i + \text{const.}, \end{aligned}$$

where $\boldsymbol{r}_i = \boldsymbol{b} - A\boldsymbol{x}_i$. Thus, the function $Q(\boldsymbol{x})$ is minimized at

$$\alpha_i = \frac{\boldsymbol{p}_i^T \boldsymbol{r}_i}{\boldsymbol{p}_i^T A \boldsymbol{p}_i}. \tag{4.15}$$

Or, α_i is also determined by the A-orthogonal condition of Eq. (4.6).

$$(\hat{\boldsymbol{x}} - \boldsymbol{x}_{i+1}, \boldsymbol{p}_i)_A = (\hat{\boldsymbol{x}} - \boldsymbol{x}_i - \alpha_i \boldsymbol{p}_i, \boldsymbol{p}_i)_A = (\boldsymbol{r}_i, \boldsymbol{p}_i) - \alpha_i (\boldsymbol{p}_i, A\boldsymbol{p}_i) = 0. \tag{4.16}$$

Hence $\alpha_i = \frac{(\boldsymbol{p}_i, \boldsymbol{r}_i)}{(\boldsymbol{p}_i, A\boldsymbol{p}_i)}$. Note that $\hat{\boldsymbol{x}} - \boldsymbol{x}_{i+1} = \{(\hat{\boldsymbol{x}} - \boldsymbol{x}_i) - \alpha_i \boldsymbol{p}_i\} \perp_A \boldsymbol{p}_j$, $j < i$.

From now on, consider how to generate $\boldsymbol{p}_i$. First, since $\boldsymbol{p}_0 \in K_1(A, \boldsymbol{r}_0) = \text{Span}\{\boldsymbol{r}_0\}$, then $\boldsymbol{p}_0 = \boldsymbol{r}_0$. Next, assume $\{\boldsymbol{p}_0,\ \boldsymbol{p}_1,\ \ldots,\ \boldsymbol{p}_i\}$ to be an A-orthogonal base of $K_{i+1}(A, \boldsymbol{r}_0)$. From Eq. (4.8), (4.10) and (4.14), $\boldsymbol{p}_{i+1}$ is determined using $\boldsymbol{r}_{i+1}$

$$\boldsymbol{p}_{i+1} = \boldsymbol{r}_{i+1} - \beta_0 \boldsymbol{p}_0 - \cdots - \beta_i \boldsymbol{p}_i. \tag{4.17}$$

β_j is obtained by the A-orthogonal condition: $(\boldsymbol{p}_{i+1}, A\boldsymbol{p}_j) = 0,\ j \le i$, hence

$$\beta_j = \frac{(\boldsymbol{r}_{i+1}, A\boldsymbol{p}_j)}{(\boldsymbol{p}_j, A\boldsymbol{p}_j)}. \tag{4.18}$$

Since $A\boldsymbol{p}_j \in K_{j+1}(A, \boldsymbol{r}_0)$, for $j+1 < i+1$

$$(\boldsymbol{r}_{i+1}, A\boldsymbol{p}_j) = 0, \tag{4.19}$$

thus Eq. (4.17) follows the recurrence

$$\boldsymbol{p}_{i+1} = \boldsymbol{r}_{i+1} - \beta_i \boldsymbol{p}_i. \tag{4.20}$$

Now $\{\boldsymbol{p}_0,\ \boldsymbol{p}_1,\ \ldots,\ \boldsymbol{p}_{i+1}\}$ is an A-orthogonal base of $K_{i+2}(A, \boldsymbol{r}_0)$. Using Eq. (4.12) and (4.20), A-orthogonal base $\{\boldsymbol{p}_0,\ \boldsymbol{p}_1,\ \ldots,\ \boldsymbol{p}_{N-1}\}$ of $K_N(A, \boldsymbol{r}_0)$ can be iteratively constructed.

Consequently, the CG algorithm for the solution of $A\boldsymbol{x} = \boldsymbol{b}$ can be represented by Figure 4.2. The iteration stops if $\boldsymbol{x}_i$ is accurate enough. Since the solution is not given, this is tested by the

$i = 0$;

Let $\boldsymbol{x}_i$ be an initial approximation.

$\boldsymbol{p}_i = \boldsymbol{r}_i = \boldsymbol{b} - A\boldsymbol{x}_i$;

while (1) {

$\alpha_i = (\boldsymbol{p}_i, \boldsymbol{r}_i)/(\boldsymbol{p}_i, A\boldsymbol{p}_i)$;

$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i$;

$\boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i A\boldsymbol{p}_i$;

if (convergent) break;

$\beta_i = (\boldsymbol{r}_{i+1}, A\boldsymbol{p}_i)/(\boldsymbol{p}_i, A\boldsymbol{p}_i)$;

$\boldsymbol{p}_{i+1} = \boldsymbol{r}_{i+1} + \beta_i \boldsymbol{p}_i$;

i++;

}

Figure 4.2: The CG method (1)

norm of the residual $\|\boldsymbol{r}_i\| = \|\boldsymbol{b} - A\boldsymbol{x}_i\|$. The following stopping criteria,

$$\frac{\|\boldsymbol{r}_i\|}{\|\boldsymbol{r}_0\|} \leq \varepsilon \tag{4.21}$$

is usually used, where $\varepsilon \simeq 10^{-16}$ in IEEE double precision computation.

4.1.1 The finite termination property

The CG method minimizes $Q(\boldsymbol{x}) = \frac{1}{2}\|\hat{\boldsymbol{x}} - \boldsymbol{x}\|_A^2 + C$ over $\boldsymbol{x}_0 + K_i(A, \boldsymbol{r}_0)$, thus it is optimal. From Eq. (4.5), if $\hat{\boldsymbol{x}} \in \boldsymbol{x}_0 + K_i(A, \boldsymbol{r}_0)$, the iteration of the CG method terminates at most after i iterations. Since A is nonsingular, it solves the linear equation $A\boldsymbol{x} = \boldsymbol{b}$ exactly at most after N iterations. In fact, it is originally proposed as a direct solution by Hestenes and Stiefel [26], unfortunately, this discussion is only true in exact arithmetic since the orthogonal condition does not exactly hold because of rounding-off error.

4.1.2 Computational consideration

Using the A-orthogonal property and the orthogonal property, computational cost of α_i and β_i can be reduced.

Since $\boldsymbol{r}_{i+1} \perp K_i(A, \boldsymbol{r}_0)$ and $\boldsymbol{p}_i \in K_i(A, \boldsymbol{r}_0)$,

$$(\boldsymbol{r}_{i+1}, \boldsymbol{p}_i) = 0. \tag{4.22}$$

Let $\boldsymbol{x}$ be an initial approximation.

$\boldsymbol{p} = \boldsymbol{r} = \boldsymbol{b} - A\boldsymbol{x}$;

$r_0 = (\boldsymbol{r}, \boldsymbol{r})$;

while (1) {

    $\boldsymbol{q} = A\boldsymbol{p}$;

    $\alpha = r_0/(\boldsymbol{p}, \boldsymbol{q})$;

    $\boldsymbol{x} = \boldsymbol{x} + \alpha\boldsymbol{p}$;

    $\boldsymbol{r} = \boldsymbol{r} - \alpha\boldsymbol{q}$;

    if (convergent) break;

    $r_1 = (\boldsymbol{r}, \boldsymbol{r})$;

    $\beta = r_1/r_0$;

    $\boldsymbol{p} = \boldsymbol{r} + \beta\boldsymbol{p}$;

    $r_0 = r_1$;

}

Figure 4.3: The CG method (2)

From Eq. (4.12),

$$(\boldsymbol{p}_{i+1}, \boldsymbol{r}_{i+1}) = (\boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+1}). \tag{4.23}$$

Thus

$$\alpha_i = \frac{(\boldsymbol{p}_i, \boldsymbol{r}_i)}{(\boldsymbol{p}_i, A\boldsymbol{p}_i)} = \frac{(\boldsymbol{r}_i, \boldsymbol{r}_i)}{(\boldsymbol{p}_i, A\boldsymbol{p}_i)} \tag{4.24}$$

From Eq. (4.12) and (4.24),

$$\begin{aligned}(\boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+1}) &= -\alpha_i(A\boldsymbol{p}_i, \boldsymbol{r}_{i+1}) \\ &= -\alpha_i(A\boldsymbol{p}_i, \boldsymbol{p}_{i+1}) + \alpha_i\beta_i(A\boldsymbol{p}_i, \boldsymbol{p}_i) \\ &= \beta_i(\boldsymbol{r}_i, \boldsymbol{r}_i)\end{aligned}$$

Thus,

$$\beta_i = \frac{(\boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+1})}{(\boldsymbol{r}_i, \boldsymbol{r}_i)} \tag{4.25}$$

In consequence, the CG algorithm saving computational cost is represented by Figure 4.3. Updating vectors $\boldsymbol{x}_i$, $\boldsymbol{p}_i$ and $\boldsymbol{r}_i$ needs only the previous vectors, so these vectors can be overwritten.

4.1.3 Three-term recurrence and Lanczos polynomial

As described in Section 4.1, the iteration of the CG method constructs an orthogonal base

$$\{\boldsymbol{r}_0,\ \boldsymbol{r}_1,\ \ldots,\ \boldsymbol{r}_i\} \tag{4.26}$$

of $K_i(A, \boldsymbol{r}_0)$. From Eq. (4.12) and (4.20), the following recurrence

$$\begin{aligned} \boldsymbol{r}_1 &= \boldsymbol{r}_0 - \alpha_0 A\boldsymbol{r}_0 & (4.27)\\ \boldsymbol{r}_{i+2} &= \left(1 + \frac{\alpha_{i+1}}{\alpha_i}\beta_i\right)\boldsymbol{r}_{i+1} - \frac{\alpha_{i+1}}{\alpha_i}\beta_i\boldsymbol{r}_i - \alpha_{i+1}A\boldsymbol{r}_{i+1} \qquad (i \geq 0) & (4.28) \end{aligned}$$

is obtained. It follows

$$\boldsymbol{r}_i = P_i(A)\boldsymbol{r}_0, \tag{4.29}$$

where $P_i(A)$ called the *Lanczos* polynomial [35] is a polynomial of degree i. Note that $P_i(0) = 1$. This residual polynomial characterizes the CG method. Let $\boldsymbol{e}_i = \hat{\boldsymbol{x}} - \boldsymbol{x}_i$,

$$\boldsymbol{e}_i = P_i(A)\boldsymbol{e}_0 \tag{4.30}$$

also holds, since $\boldsymbol{r}_i = A\boldsymbol{e}_i$.

From the optimality of the CG method, this polynomial minimizes $Q(\boldsymbol{x}_i) = \frac{1}{2}\|\hat{\boldsymbol{x}} - \boldsymbol{x}_i\|_A + C = \frac{1}{2}\|\boldsymbol{e}_i\|_A + C$ where $\boldsymbol{x}_i \in \boldsymbol{x}_0 + K_i(A, \boldsymbol{r}_0)$ and C is a constant. This means

$$\begin{aligned} \|\boldsymbol{e}_i\|_A &= \|P_i(A)\boldsymbol{e}_0\|_A \\ &= \min_{R_i \in \pi_i^1} \|R_i(A)\boldsymbol{e}_0\|_A, & (4.31) \end{aligned}$$

where π_i^1 is the set of polynomials of degree i normalized such that $R_i(0) = 1$.

4.2 Rate of Convergence of the CG Method

As described in the previous section, the error of the CG method $\boldsymbol{e}_k = P_k(A)\boldsymbol{e}_0$ of the kth iteration is decided such that

$$\|\boldsymbol{e}_k\|_A = \min_{P_k \in \pi_k^1} \|P_k(A)\boldsymbol{e}_0\|_A, \tag{4.32}$$

where π_k^1 is the set of polynomials of degree k normalized such that $P_k(0) = 1$. Let $\boldsymbol{\nu}_i$ be eigenvector of A and λ_i be the corresponding eigenvalue,

$$\begin{aligned} \min_{P_k \in \pi_k^1} \|P_k(A)\boldsymbol{e}_0\|_A &= \min_{P_k \in \pi_k^1} \|\sum_{i=0}^{N-1} P_k(\lambda_i)c_i\boldsymbol{\nu}_i\|_A \\ &\leq \min_{P_k \in \pi_k^1} \max_{\lambda \in S(A), \lambda \neq 0} |P_k(\lambda)|\ \|\boldsymbol{e}_0\|_A, & (4.33) \end{aligned}$$

where $\boldsymbol{e}_0 = \sum_{i=0}^{N-1} c_i \boldsymbol{\nu}_i$ and $S(A)$ is the spectrum of the matrix A. Convergence factor is given as

$$\frac{\|\boldsymbol{e}_k\|_A}{\|\boldsymbol{e}_0\|_A} \leq \min_{P_k \in \pi_k^1} \max_{\lambda \in S(A), \lambda \neq 0} |P_k(\lambda)|.$$

Even if $S(A)$ is exactly given, it is necessary for estimating the convergence factor to solve the best approximation problem on the discrete set of points, which is quite troublesome. Thus this problem is replaced with the best approximation problem with the interval. When there are not isolated eigenvalues, this approximation gives a quite good estimate of the convergence factor.

$$\min_{P_k \in \pi_k^1} \max_{\lambda \in S(A), \lambda \neq 0} |P_k(\lambda)| \leq \min_{P_k \in \pi_k^1} \max_{\lambda_{\min} \leq \lambda \leq \lambda_{\max}} |P_k(\lambda)| \tag{4.34}$$

Here, let $a = \lambda_{\min}$ and $b = \lambda_{\max}$. P_k is represented with T_k the Chebyshev polynomial,

$$\min_{P_k \in \pi_k^1} \max_{a \leq \lambda \leq b} |P_k(\lambda)| = \max \frac{|T_k\left(\frac{b+a-2x}{b-a}\right)|}{T_k\left(\frac{b+a}{b-a}\right)}, \tag{4.35}$$

where

$$T_k(x) = \frac{1}{2}\left\{(x + \sqrt{x^2-1})^k + (x - \sqrt{x^2-1})^k\right\}. \tag{4.36}$$

Since $\max |T_k\left(\frac{b+a-2x}{b-a}\right)| = 1$,

$$\min_{P_k \in \pi_k^1} \max_{a \leq \lambda \leq b} |P_k(\lambda)| = \frac{2c^k}{1+c^{2k}}, \tag{4.37}$$

where $c = \frac{\sqrt{\sigma}-1}{\sqrt{\sigma}+1}$ and $\sigma = \frac{b}{a}$ is the condition number of A. Thus

$$\frac{\|\boldsymbol{e}_k\|_A}{\|\boldsymbol{e}_0\|_A} \leq \frac{2c^k}{1+c^{2k}}. \tag{4.38}$$

Therefore the average convergence factor is estimated by $c\left(\frac{2}{1+c^{2k}}\right)^{\frac{1}{k}}$ and the factor approaches to c as $k \to \infty$. Asymptotic rate of convergence is also estimated by $(-\log_{10} c)$.

When stopping condition is

$$\frac{\|\boldsymbol{e}_k\|_A}{\|\boldsymbol{e}_0\|_A} \leq \varepsilon, \tag{4.39}$$

let k be the number of iterations until convergence,

$$\frac{2c^k}{1+c^{2k}} \leq \varepsilon. \tag{4.40}$$

Thus

$$k \geq \log_{\frac{1}{c}} \frac{1+\sqrt{1-\varepsilon^2}}{\varepsilon}. \tag{4.41}$$

4.3 Preconditioned CG Method

As described in the previous section, the convergence factor of the CG method strongly depends on the distribution of eigenvalues of A. When there are not isolated eigenvalues, the asymptotic

$i = 0$;

Let $\tilde{\boldsymbol{x}}_i$ be an initial approximation.

$\tilde{\boldsymbol{p}}_i = \tilde{\boldsymbol{r}}_i = \tilde{\boldsymbol{b}} - \tilde{A}\tilde{\boldsymbol{x}}_i$;

while (1) {

$\alpha_i = (\tilde{\boldsymbol{r}}_i, \tilde{\boldsymbol{r}}_i)/(\tilde{\boldsymbol{p}}_i, \tilde{A}\tilde{\boldsymbol{p}}_i)$;

$\tilde{\boldsymbol{x}}_{i+1} = \tilde{\boldsymbol{x}}_i + \alpha_i \tilde{\boldsymbol{p}}_i$;

$\tilde{\boldsymbol{r}}_{i+1} = \tilde{\boldsymbol{r}}_i - \alpha_i \tilde{A}\tilde{\boldsymbol{p}}_i$;

if (convergent) break;

$\beta_i = (\tilde{\boldsymbol{r}}_{i+1}, \tilde{\boldsymbol{r}}_{i+1})/(\tilde{\boldsymbol{r}}_i, \tilde{\boldsymbol{r}}_i)$;

$\tilde{\boldsymbol{p}}_{i+1} = \tilde{\boldsymbol{r}}_{i+1} + \beta_i \tilde{\boldsymbol{p}}_i$;

$i{+}{+}$;

}

Figure 4.4: The PCG method (1)

convergence factor of the CG method is estimated by

$$\frac{\sqrt{\sigma} - 1}{\sqrt{\sigma} + 1}, \tag{4.42}$$

where σ is the condition number of A. To improve the convergence factor, preconditioning technique is quite useful. Let U be a nonsingular matrix, the linear equation $A\boldsymbol{x} = \boldsymbol{b}$ is transformed into $\tilde{A}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}}$, where $\tilde{A} = UAU^T$, $\tilde{\boldsymbol{x}} = U^{-T}\boldsymbol{x}$ and $\tilde{\boldsymbol{b}} = U\boldsymbol{b}$. From Eq. (4.42), the convergence factor is improved as $\sigma \to 1$. In the extremely case $\tilde{A} = I$,

$$U^T U = A^{-1}. \tag{4.43}$$

Hence, when $U^T U \simeq A^{-1}$, the convergence factor is improved. The CG algorithm for $\tilde{A}\tilde{\boldsymbol{x}} = \tilde{\boldsymbol{b}}$, which is called the *preconditioned CG* (PCG) method, is represented by Figure 4.4. To implement the algorithm of Figure 4.4, the matrix-vector multiplication of the preconditioned matrix $\tilde{A}$ is necessary, however it is costly. Fortunately this multiplication can be avoided.

$$\begin{aligned} \tilde{\boldsymbol{r}} &= \tilde{\boldsymbol{b}} - \tilde{A}\tilde{\boldsymbol{x}} \\ &= U\boldsymbol{b} - UAU^T U^{-T}\boldsymbol{x} = U\boldsymbol{r} \end{aligned} \tag{4.44}$$

From Eq. (4.44),

$$(\tilde{\boldsymbol{r}}, \tilde{\boldsymbol{r}}) = (U\boldsymbol{r}, U\boldsymbol{r}) = (U^T U\boldsymbol{r}, \boldsymbol{r}), \tag{4.45}$$

and since $\tilde{\boldsymbol{r}}_{i+1} = \tilde{\boldsymbol{r}}_i - \alpha_i \tilde{A}\tilde{\boldsymbol{p}}_i$,

$$\boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i AU^T \tilde{\boldsymbol{p}}_i. \tag{4.46}$$

$i = 0$;

Let $\boldsymbol{x}_i$ be an initial approximation.

$\boldsymbol{r}_i = \boldsymbol{b} - A\boldsymbol{x}_i$;

$\bar{\boldsymbol{p}}_i = \bar{\boldsymbol{r}}_i = M\boldsymbol{r}_i$;

while (1) {

$\bar{\boldsymbol{q}}_i = A\bar{\boldsymbol{p}}_i$;

$\alpha_i = (\bar{\boldsymbol{r}}_i, \boldsymbol{r}_i)/(\bar{\boldsymbol{p}}_i, \bar{\boldsymbol{q}}_i)$;

$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i\bar{\boldsymbol{p}}_i$;

$\boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i\bar{\boldsymbol{q}}_i$;

if (convergent) break;

$\bar{\boldsymbol{r}}_{i+1} = M\boldsymbol{r}_{i+1}$;

$\beta_i = (\bar{\boldsymbol{r}}_{i+1}, \boldsymbol{r}_{i+1})/(\bar{\boldsymbol{r}}_i, \boldsymbol{r}_i)$;

$\bar{\boldsymbol{p}}_{i+1} = \bar{\boldsymbol{r}}_{i+1} + \beta_i\bar{\boldsymbol{p}}_i$;

$i{+}{+}$;

}

Figure 4.5: The PCG method (2)

Furthermore,

$$(\tilde{\boldsymbol{p}}, \tilde{A}\tilde{\boldsymbol{p}}) = (\tilde{\boldsymbol{p}}, UAU^T\tilde{\boldsymbol{p}}) = (U^T\tilde{\boldsymbol{p}}, AU^T\tilde{\boldsymbol{p}}), \tag{4.47}$$

and

$$U^T\tilde{\boldsymbol{p}}_{i+1} = U^TU\boldsymbol{r}_i + \beta_iU^T\tilde{\boldsymbol{p}}_i. \tag{4.48}$$

Thus let $M = U^TU$, $\bar{\boldsymbol{r}}_i = M\boldsymbol{r}_i$ and $\bar{\boldsymbol{p}}_i = U^T\tilde{\boldsymbol{p}}_i$, the PCG method without matrix-vector multiplication of the preconditioned matrix is represented by Figure 4.5. Because the same matrix-vector multiplication $A\bar{\boldsymbol{p}}_i$ is necessary twice, it is replaced by $\bar{\boldsymbol{q}}_i = A\bar{\boldsymbol{p}}_i$. To keep the readability, there are two $(\bar{\boldsymbol{r}}_i, \boldsymbol{r}_i)$ and $(\bar{\boldsymbol{r}}_{i+1}, \boldsymbol{r}_{i+1})$ in Figure 4.5, of course, it is enough to compute $(\bar{\boldsymbol{r}}_{i+1}, \boldsymbol{r}_{i+1})$ at the ith iteration, since $(\bar{\boldsymbol{r}}_i, \boldsymbol{r}_i)$ has already computed at the previous $(i-1)$th iteration. It is to be noted that even though the PCG method of Figure 4.5 does not explicitly utilize $\tilde{A}$, $\tilde{\boldsymbol{r}}$ and $\tilde{\boldsymbol{x}}$, this algorithm minimizes $Q(\tilde{\boldsymbol{x}}_i) = \frac{1}{2}\tilde{\boldsymbol{x}}_i^T\tilde{A}\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{b}}^T\tilde{\boldsymbol{x}}_i$ in Krylov subspace $K_i(\tilde{A}, \tilde{\boldsymbol{r}}_0)$. Moreover, because $M = U^TU$, M should be symmetric and positive definite.

```
for (i = 0; i < n; i ++) {
    for (j = 0; j < i; j ++)
        if ((i, j) in S) {
            L[i][j] = A[i][j];
            for (k = 0; k < j; k ++)
                L[i][j] -= L[i][k] * L[k][k] * L[j][k];
            L[i][j] /= L[j][j];
        }
        else
            L[i][j] = 0;
    L[i][i] = A[i][i];
    for (k = 0; k < i; k ++)
        L[i][i] -= L[i][k] * L[i][k] * L[k][k];
}
```

Figure 4.6: Incomplete Cholesky decomposition

4.3.1 ICCG method

Let $A = [a_{ij}]$ be an s.p.d. matrix, the decomposition $A = LL^T$ is called the *Cholesky decomposition*, where L is a lower triangular matrix, however L may be dense even if A is sparse. The idea of incomplete factorization methods is to reject those fill-in entries outside a priori or adaptively (dynamically) chosen sparsity pattern. An incomplete factorization $A = LL^T - R$ is called the *incomplete Cholesky decomposition*, whose existence and uniqueness has been proved for M-matrix by Meijerink and van der Vorst [37].

Because LL^T is s.p.d., $(LL^T)^{-1}$ is also s.p.d., thus the incomplete Cholesky factorization can be used as the preconditioner of the CG method. The CG method with the incomplete Cholesky decomposition preconditioner is called the *ICCG* method [37, 31].

When a sparsity pattern $\mathcal{S}$ is given a priori, the incomplete Cholesky decomposition $A = LL^T - R$, which is called factorization *by position*, is described by Figure 4.6. If $(i,\ j) \in \mathcal{S}$, the fill-in entry is accepted. On the other hand, the sparsity set can be given dynamically during the factorization, which is called factorization *by value* [6]. In this case, fill-in entries that are sufficiently small to satisfy

$$|a_{ij}^{(r+1)}| \leq c|a_{ii}^{(r+1)} a_{jj}^{(r+1)}|^{\frac{1}{2}}, \tag{4.49}$$

are neglected, where $a_{ij}^{(r+1)}$ is $(i,\ j)$ element at the $(r+1)$th elimination stage and $0 \leq c \leq 1$.

A common choice of the sparsity set is to let

$$\mathcal{S} = \mathcal{S}_0 = \{(i,j) \mid a_{ij} \neq 0\}. \tag{4.50}$$

According to Gustafsson [18], the sparsity set of order q is defined in the following way. Let $\mathcal{R}_q$ be the sparsity set that is defined by the position of the fill-in entries of the incomplete factorization based on the sparsity set $\mathcal{S}_q$, and let $\mathcal{S}_{q+1} = \mathcal{S}_q \cap \mathcal{R}_q$ $(q = 0,\ 1,\ \ldots)$.

Discretizing two-dimensional rectangular computational domain into $n \times n$ lexicographically numbered grid by the standard five-point difference approximation, a block tridiagonal matrix A arises from second-order elliptic partial differential equation, whose sparsity set is represented by

$$\mathcal{S} = \mathcal{S}_0 = \{(i,\ j) \mid i = j,\ j \pm 1,\ j \pm n\}. \tag{4.51}$$

The sparsity set of order 1 for A is denoted by

$$\mathcal{S}_1 = \{(i,\ j) \mid i = j,\ j \pm 1,\ j \pm n,\ j \pm n \mp 1\}. \tag{4.52}$$

The CG method with this incomplete factorization is called the *ICCG(1)* or *ICCG(1,2)* method.

The incomplete factorization can be modified in various way: adding the deleted entries to the diagonal entries in the same row, or modifying the pivot entry by adding a positive (small) number to it. For further study, see Gustafsson [18, 17], Axelsson and Lindskog [4] and Axelsson [7].

The ICCG method needs the solution of lower and upper triangular matrices, which is not readily vectorizable or parallelizable. Johnson and Paul [30] and van der Vorst [54] have proposed some variants for vector machines. These variants have ample data parallelism, however these variants have cheaper rate of convergence. The IC preconditioning can be vectorized by loop restructuring, which is implemented using indirect index (or list vector) in FORTRAN, however its vector length is short.

4.3.2 SCG method

The SCG method is a CG method with diagonal scaling preconditioner. Let D be a diagonal matrix containing diagonal elements of A. This method is described by Figure 4.7. For efficient implementation of the SCG method, $\bar{\boldsymbol{r}}$ in Figure 4.5 is not generally used since memory store operation is more costly than the diagonal scaling, which is a merit of the SCG method, because storage for $\bar{\boldsymbol{r}}$ is dispensable.

The diagonal scaling preconditioner does not need any communication, thus the SCG method is 100% vectorizable and parallelizable, then the computational time for one iteration becomes

$i = 0$;

while (!convergence) {

$\alpha_i = (D^{-1}\boldsymbol{r}_i, \boldsymbol{r}_i)/(\boldsymbol{p}_i, A\boldsymbol{p}_i)$;

$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha_i \boldsymbol{p}_i$;

$\boldsymbol{r}_{i+1} = \boldsymbol{r}_i - \alpha_i A\boldsymbol{p}_i$;

convergence test;

$\beta_i = (D^{-1}\boldsymbol{r}_{i+1}, \boldsymbol{r}_{i+1})/(D^{-1}\boldsymbol{r}_i, \boldsymbol{r}_i)$;

$\boldsymbol{p}_{i+1} = D^{-1}\boldsymbol{r}_{i+1} + \beta_i \boldsymbol{p}_i$;

$i{+}{+}$;

}

Figure 4.7: The SCG method

shorter proportionally to the number of processors. Though the SCG method has cheaper rate of convergence than that of the ICCG method, it has been reported that total convergence time of the SCG method can be shorter than that of the ICCG method on vector machines, since the time for one iteration of the ICCG method is not reduced so much [23].

Chapter 5

MULTIGRID METHOD

The chapter explains the multigrid (MG) method. There are many good references; Stüben and Trottenberg [51], Hackbusch [21] and Wesseling [59].

Section 5.3 shows the mesh-independent convergence property of the two-grid method and in Section 5.4, a smoothing factor that shows a practical convergence factor of the multigrid method is explained.

5.1 Two-grid Method

The simplest form of the MG method is two-grid method which exploits only two grids: a fine grid Ω_2 and a coarse grid Ω_1. The subscript is referred to as the *grid level* or the *grid number*. Usually, $\Omega_1 \subset \Omega_2$ and the mesh size h_1 on Ω_1 is double the mesh size h_2 on Ω_2, however, it is not a necessary condition. Consider a system of linear equations on the find grid Ω_2

$$L_2 \boldsymbol{x}_2 = \boldsymbol{f}_2. \tag{5.1}$$

The subscript of A, $\boldsymbol{x}$ and $\boldsymbol{b}$ means the grid level. This subscript may be omitted in this thesis if it is clear.

The two-grid method consists of a smoothing step and a coarse-grid correction step. In the smoothing step, high-frequency (or rough) components of the residual decay by a smoother on the fine grid. The smoother is generally ν_1 iterations of an iterative method that has the *smoothing property* [20]. The smoothing property is a sufficient condition of the smoother such that two-grid (or MG) method converges independently of the mesh size, which will be explained in Section 5.3. In the coarse-grid correction step, the residual after the smoothing on the fine grid, which is expected to contain low-frequency (or smooth) components, is transfered to the coarse grid. This

```
while (!convergence) {
    x_2 = pre_smoother(L_2, f_2, x_2, ν_1);      // pre-smoothing
    r_1 = restrict(f_2 − L_2 x_2);               // coarse-grid correction
    Solve L_1 d_1 = r_1;
    x_2 = x_2 + prolongate(d_1);
    x_2 = post_smoother(L_2, f_2, x_2, ν_2);     // post-smoothing
}
```

Figure 5.1: The two-grid method

operation is called the *restriction*. Then solve

$$L_1 \boldsymbol{d}_1 = \boldsymbol{r}_1 \tag{5.2}$$

accurately enough, where $\boldsymbol{r}_1$ is the restricted residual. The solution $\boldsymbol{d}_1$ is interpolated to the fine grid. This operation is called the *prolongation* or the *interpolation*. Then the approximation after the smoothing is corrected by this interpolated solution.

One iteration of the two-grid method consists of ν_1 iterations of the pre-smoother, the coarse-grid correction and ν_2 iterations of the post-smoother. The two-grid algorithm is depicted by Figure 5.1. The smoothing before the coarse-grid correction is called the *pre-smoothing* and that after the coarse-grid correction is called the *post-smoothing*. Pre_smoother(L_2, $\boldsymbol{f}_2$, $\boldsymbol{x}_2$, ν_1) in Figure 5.1 means ν_1 iterations of the pre-smoother with $\boldsymbol{x}_2$ an initial approximation.

5.1.1 Restriction and prolongation

The domain of the partial differential equation to be solved is assumed to be a d-dimensional unit cube for simplicity. This assumption is not a serious limitation, since the current main trend in grid generation consists of decomposition of the physical domain in subdomains, each of which is mapped onto a cubic computational domain. In one dimension, a computational grid in $(0,\ 1)$ is defined by

$$\Omega_2 = \{x \in (0,\ 1) : x = jh_2,\ j = 1,\ 2,\ \ldots,\ N_2 - 1,\ h_2 = \frac{1}{N_2}\}, \tag{5.3}$$

with N_2 meshes, which is the fine grid. The coarse grid is usually defined by

$$\Omega_1 = \{x \in (0,\ 1) : x = jh_1,\ j = 1,\ 2,\ \ldots,\ N_1 - 1,\ h_1 = \frac{1}{N_1}\}, \tag{5.4}$$

where $N_1 = \frac{N_2}{2}$. The two grids are depicted by Figure 5.2 for $N_2 = 8$. For simplicity, N_2 is

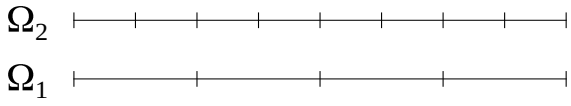

Figure 5.2: The fine grid and the coarse grid in 1 dimension

assumed even. The standard prolongation p by linear interpolation is given by

$$p\boldsymbol{u}_1(x) = \begin{cases} \boldsymbol{u}_1(x), & \text{if } x \in \Omega_1 \\ \frac{\boldsymbol{u}_1(x-h_2)+\boldsymbol{u}_1(x+h_2)}{2}, & \text{otherwise} \end{cases} \tag{5.5}$$

where $x \in \Omega_2$. The prolongation is represented by a rectangular $N_2 \times N_1$ matrix

$$p = \frac{1}{2}\begin{pmatrix} 1 & & & & & & \\ 2 & & & & & & \\ 1 & 1 & & & & & \\ & 2 & & & & & \\ & & \ddots & & & & \\ & & & & 2 & & \\ & & & & 1 & 1 & \\ & & & & & 2 & \\ & & & & & 1 & \end{pmatrix}. \tag{5.6}$$

The simplest choice of a restriction is trivial injection r_{inj} defined by

$$r_{\text{inj}}\boldsymbol{u}_2(x) = \boldsymbol{u}_2(x), \tag{5.7}$$

where $x \in \Omega_1$. This restriction is easily implemented, however it has some disadvantages. The restriction r can be defined by adjoint of the prolongation,

$$r\,\boldsymbol{u}_2(x) = \frac{\boldsymbol{u}_2(x-h_2) + 2\boldsymbol{u}_2(x) + \boldsymbol{u}_2(x+h_2)}{4}, \tag{5.8}$$

where $x \in \Omega_1$. The corresponding $N_1 \times N_2$ matrix is

$$r = \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 & & & & & & & \\ & & 1 & 2 & 1 & & & & & \\ & & & & & \ddots & & & & \\ & & & & & & 1 & 2 & 1 & \\ & & & & & & & 1 & 2 & 1 \end{pmatrix} = \frac{1}{2}p^T. \tag{5.9}$$

r does not equal to p^T, but $r = p^*$ where p^* is the adjoint with respect to the scaler products

$$\langle v_l,\ u_l \rangle = h_l^d \sum_{x \in \Omega_l} v_l(x)\overline{u_l(x)}, \tag{5.10}$$

where l is the grid level. The prolongation and the restriction are usually represented by *stencil notation*. The element of a matrix A on a d-dimensional computational grid Ω is represented by

$A(i,\ j)$ where $i \in \Omega$ and $j \in \{0,\ \pm 1,\ \pm 2,\ \ldots\}^d$, which represents $(l,\ m)$-element of A where l and m are grid numbers of i and $i+j$ respectively. $S_A = \{j \mid \exists i \in \Omega \text{ with } A(i,\ j) \neq 0\}$ is called the *structure* of A which denotes the set of locations of non-zero elements. The set of values $A(i,\ j)$ with $j \in S_A$ is called the *stencil* of A of grid point i, which is denoted by $[A]_i$. For example, when A has nine-point pattern in two dimensions,

$$[A]_i = \begin{bmatrix} A(i,\ -e_1+e_2) & A(i,\ e_2) & A(i,\ e_1+e_2) \\ A(i,\ -e_1) & A(i,\ 0) & A(i,\ e_1) \\ A(i,\ -e_1-e_2) & A(i,\ -e_2) & A(i,\ e_1-e_2) \end{bmatrix}, \tag{5.11}$$

where $e_1 = (1,\ 0)$ and $e_2 = (0,\ 1)$. Using stencil notation, the restriction is denoted by

$$[r]_i = \frac{1}{4}[1\ 2\ 1], \tag{5.12}$$

where $i \in \Omega_1$. The prolongation p is not denoted by stencil notation, however a convenient way to denote p is by specifying p^T.

$$[p^T]_i = \frac{1}{2}[1\ 2\ 1]. \tag{5.13}$$

In two dimensions, the prolongation by bilinear interpolation is defined by

$$pv_1(x,y) = \begin{cases} v_1(x,y), & \text{if } (x,y) \in \Omega_1 \\ \{v_1(x-h_2,y) + v_1(x+h_2,y)\}/2, & \text{if } (x \pm h_2, y) \in \Omega_1 \\ \{v_1(x,y-h_2) + v_1(x,y+h_2)\}/2, & \text{if } (x, y \pm h_2) \in \Omega_1 \\ \{v_1(x-h_2,y-h_2) + v_1(x-h_2,y+h_2) & \\ \quad + v_1(x+h_2,y-h_2) + v_1(x+h_2,y+h_2)\}/4, & \text{otherwise} \end{cases} \tag{5.14}$$

where $(x,\ y) \in \Omega_2$. Using stencil notation, the prolongation by bilinear interpolation is defined by

$$[p^T]_i = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}, \tag{5.15}$$

where $i \in \Omega_2$. Alternatively, the prolongation by linear interpolation is defined by

$$[p^T]_i = \frac{1}{2}\begin{bmatrix} & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & \end{bmatrix}. \tag{5.16}$$

Using linear interpolation, coarse grids constructed by Galerkin coarse-grid approximation described in Subsection 5.1.3 have greater sparsity than these using bilinear interpolation, hence the computation on coarse grids is cheaper. However, the approximation may be a loss of symmetry because of rounding-off error, even though the solution of a problem has a certain symmetry, while bilinear interpolation preserves symmetry exactly.

When the partial differential equation to be solved has strongly discontinuous coefficients, linear (or bilinear) interpolation across discontinuities is inaccurate. Instead of the standard interpolation, *operator-dependent prolongation* has to be used. Operator-dependent prolongation aims to approximate the correct jump condition using information from the discrete operator, which have been proposed by Alcouffe et al. [1], Kettler and Meijerink [32], Dendy [11] and Kettler [33].

5.1.2 Accuracy condition for transfer operators

The proofs of mesh-size independent rate of convergence of the MG method assume that p and r satisfy certain conditions (Brandt [10] and Hackbusch [21]). The last author gives the following simple condition:

$$m_p + m_r > 2m \tag{5.17}$$

The necessity of Eq. (5.17) has been shown by Hemker [24]. Here orders m_p and m_r are defined as the highest degree plus one of polynomials that are interpolated exactly by p or sr^*, respectively, with s a scaling factor that can be chosen free, and $2m$ is the order of the partial differential equation to be solved.

5.1.3 Coarse-grid approximation

To construct a coarse grid matrix, there are basically two ways. Like the fine grid matrix, the coarse grid matrix is obtained by discretizing the partial differential equation on the coarse grid. This is called *discretization coarse-grid approximation* (DCA). The other is justified as follows. A system of linear equation $A\boldsymbol{x} = \boldsymbol{f}$ is equivalent to

$$(A\boldsymbol{x},\ \boldsymbol{v}) = (\boldsymbol{f},\ \boldsymbol{v}) \tag{5.18}$$

where $\boldsymbol{v}$ is any vector on the fine grid. Let $\bar{\boldsymbol{x}}$ be the solution on the coarse grid such that

$$(Ap\bar{\boldsymbol{x}},\ \tilde{p}\bar{\boldsymbol{v}}) = (\boldsymbol{f},\ \tilde{p}\bar{\boldsymbol{v}}), \tag{5.19}$$

where $\bar{\boldsymbol{v}}$ is any vector on the coarse grid and p and $\tilde{p}$ are prolongation operators. It follows

$$(\tilde{p}^* Ap\bar{\boldsymbol{x}},\ \bar{\boldsymbol{v}}) = (\tilde{p}^*\boldsymbol{f},\ \bar{\boldsymbol{v}}) \tag{5.20}$$

on the coarse grid. Thus, a coarse grid matrix $\bar{A}$ can be defined by

$$\bar{A} = \tilde{p}^* Ap. \tag{5.21}$$

Replacing $\tilde{p}^*$ by r, the approximation

$$\bar{A} = r\,Ap \tag{5.22}$$

```
Vector MG(L_l, f, x, μ_1, μ_2)
{
    if (l == 1)
        Solve L_1 x = f;
    else {
        x = pre_smoother(L_l, f, x, μ_1);
        d = restrict(f − L_l x);
        ν = initial_x;
        repeat (γ_{l−1}) ν = MG(L_{l−1}, d, ν, μ_1, μ_2);
        x = x + prolongate(ν);
        x = post_smoother(L_l, f, x, μ_2);
    }
    return x;
}
```

Figure 5.3: The multigrid method

of the coarse grid matrix is called *Galerkin coarse-grid approximation* (GCA).

By DCA, the coarse grid matrix is easily constructed and has the same sparsity pattern as the fine grid matrix. By GCA, the coarse grid matrix is automatically generated from the fine grid matrix, however, sparsity of the coarse grid matrix may be worse. For example, let a fine grid matrix be five-point pattern, the coarse grid matrix becomes nine-point pattern using bilinear prolongation and its adjoint restriction.

5.2 Multigrid Method

Let a sequence $\{\Omega_i \mid i = 1,\ 2,\ \ldots,\ K\}$ of increasingly finer grids be given, the MG method that exploits K grids is defined using the two-grid method recursively, which is represented by Figure 5.3. The order in which the grids are visited is referred to as the *MG schedule* or the *MG cycle*, which is determined by the parameter γ_l. When each γ_l is fixed in advance, the schedule is called the *fixed schedule*; when γ_l depends on intermediate computational results, it is called the *adaptive schedule*. Familiar MG schedules are the V-cycle and W-cycle for $\gamma_l = 1$ and $\gamma_l = 2,\ l = 1,\ 2,\ 3$, respectively, which is depicted by Figure 5.4. A schedule intermediate between V- and W-cycles is called the F-cycle which calls MG twice on each grid level l, first with $\gamma_l = 2$ later with $\gamma_l = 1$.

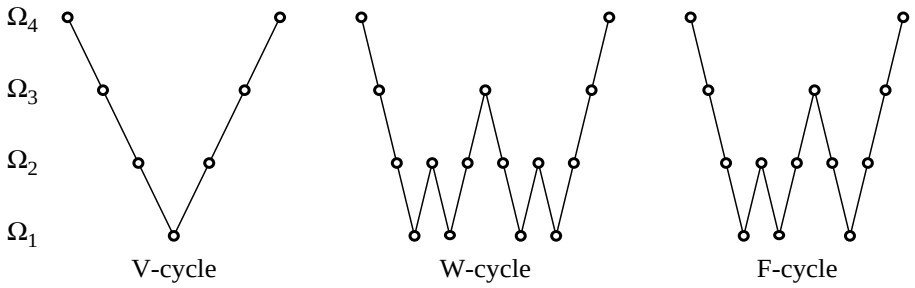

Figure 5.4: V-, W- and F-cycle diagrams

$$
\begin{aligned}
&\boldsymbol{u} = S_1^{\nu_1}\boldsymbol{u} + R_1\,\boldsymbol{f} && \text{// pre-smoothing}\\
&\boldsymbol{d} = r\,(L_2\boldsymbol{u} - \boldsymbol{f}) && \text{// coarse-grid correction}\\
&\boldsymbol{v} = L_1^{-1}\boldsymbol{d}\\
&\boldsymbol{u} = \boldsymbol{u} - p\,\boldsymbol{v}\\
&\boldsymbol{u} = S_2^{\nu_2}\boldsymbol{u} + R_2\,\boldsymbol{f} && \text{// post-smoothing}
\end{aligned}
$$

Figure 5.5: The two-grid iteration by matrix computation

5.3 Two-grid Analysis

5.3.1 Two-grid iteration matrix

The two-grid iteration of Figure 5.1 can be represented by matrix computation such as Figure 5.5. In Figure 5.5, S_1 and S_2 are iteration matrices of pre- and post-smoothers, respectively. From Figure 5.5, the two-grid iteration is denoted by

$$\boldsymbol{u} = S_2^{\nu_2}(I - pL_1^{-1}rL_2)S_1^{\nu_1}\boldsymbol{u} + \{S_2^{\nu_2}R_1 + R_2 + S_2^{\nu_2}pL_1^{-1}r(I - L_2R_1)\}\boldsymbol{f} \tag{5.23}$$

Hence the two-grid iteration matrix is

$$Q = S_2^{\nu_2}(I - pL_1^{-1}rL_2)S_1^{\nu_1} \tag{5.24}$$

or

$$Q = S_2^{\nu_2}(L_2^{-1} - pL_1^{-1}r)L_2S_1^{\nu_1}. \tag{5.25}$$

5.3.2 Two-grid rate of convergence

For simplicity, consider only for $\nu_2 = 0$. See Hackbusch [21] for $\nu_2 \neq 0$. The iteration matrix of the two-grid method is

$$T = (L_2^{-1} - pL_1^{-1}r)(LS_1^{\nu_1}), \tag{5.26}$$

so that

$$\|T\| \leq \|L_2^{-1} - pL_1^{-1}r\|\|L_2 S_1^{\nu_1}\|. \tag{5.27}$$

The separate study of the two factors in Eq. (5.27) leads to the following definitions [21].

Definition 5.3.1 (The smoothing property) S has the smoothing property if there exist a constant C_S and a function $\eta(\nu)$ independent of the mesh size such that

$$\|LS^\nu\| \leq C_S h^{-2m}\eta(\nu), \quad \eta(\nu) \to 0 \quad \text{for} \quad \nu \to \infty, \tag{5.28}$$

where $2m$ is the order of the partial differential equation to be solved.

Definition 5.3.2 (The approximation property) The approximation property holds if there exists a constant C_A independent of the mesh size such that

$$\|L^{-1} - p\bar{L}^{-1}r\| \leq C_A h^{2m}, \tag{5.29}$$

where $2m$ is the order of the partial differential equation to be solved and $\bar{L}$ is the coarse grid matrix.

If these two properties hold, mesh-independent rate of convergence of the two-grid method (with $\nu_2 = 0$) follows easily.

Theorem 5.3.1 *Let the smoothing property and the approximation property hold. Then there exists a number $\bar{\nu}$ independent of the mesh size such that*

$$\|T\| \leq C_S C_A \eta(\nu) < 1, \quad \forall \nu \geq \bar{\nu} \tag{5.30}$$

Proof. From Eq. (5.27), (5.28) and (5.29),

$$\|T\| \leq C_S C_A \eta(\nu). \tag{5.31}$$

According to the smoothing property, $\bar{\nu}$ is chosen independent of the mesh size such that the theorem holds. □

5.3.3 Smoothing property

When L is a discretization of a partial differential operator of order $2m$, Gerschgorin's theorem gives in this case

$$\|P\| \leq C_P h^{-2m} \tag{5.32}$$

with C_P some constant.

Sufficient conditions for the smoothing property are given by the following theorem [60].

Theorem 5.3.2 (Wittum 1989) *Let L and P be symmetric and positive definite, and let P satisfy Eq. (5.32). Suppose furthermore that the eigenvalues of $S(= I - P^{-1}L)$ satisfy*

$$\lambda(S) \geq -\theta > -1. \tag{5.33}$$

Then, the smoothing property holds with $C_S = C_P$ and

$$\eta(\nu) = \eta_\theta(\nu) = \max\{\nu^\nu/(\nu+1)^{\nu+1},\ \theta^\nu(1+\theta)\} \tag{5.34}$$

Proof. Since $P^{\frac{1}{2}}SP^{-\frac{1}{2}}$ is symmetric, $\lambda(S)$ is real. We can write $LS^\nu = P^{\frac{1}{2}}(I-X)X^\nu P^{\frac{1}{2}}$ with $X = P^{-\frac{1}{2}}TP^{-\frac{1}{2}}$, so that $\|LS^\nu\| \leq \|P\|\|(I-X)X^\nu\|$. X is symmetric and has the same spectrum as S. Hence Eq. (5.33) gives $\lambda(X) \geq -\theta$. Furthermore, $X - I = -P^{-\frac{1}{2}}LP^{-\frac{1}{2}}$, so that $X - I$ is negative definite. Hence, $-\theta \leq \lambda(X) < 1$, so that $\|(I-X)X^\nu\| \leq \max_{-\theta\leq x\leq 1}|(1-x)x^\nu| = \eta_\theta(\nu)$. The proof is complete by using Eq. (5.32). □

Not every convergent method satisfy Eq. (5.33). By introducing damping, every convergent method can, however, be made to satisfy Eq. (5.33), as noted by Wittum [60]. This is easily seen as follows. Let the conditions of Theorem 5.3.2 be satisfied, except Eq. (5.33), and let S be convergent. $\lambda(S)$ is real as seen in the preceding proof, and $\lambda(P^{-1}L) = 1 - \lambda(S)$; thus we have $\lambda(P^{-1}L) < 2$. Let

$$0 \leq \omega \leq \omega_\theta = \frac{1+\theta}{2}. \tag{5.35}$$

Then we have for the smallest eigenvalue of $S_\omega = I - \omega P^{-1}L$:

$$\lambda_{\min}(S_\omega) = 1 - \omega\lambda_{\max}(P^{-1}L) \geq 1 - (1+\theta) = -\theta, \tag{5.36}$$

so that S_ω satisfies Eq. (5.33).

5.4 Smoothing Analysis

Using two-grid analysis described in the previous section, mesh-independent convergence rate of the two-grid method can be proved. However, a practical convergence factor of the two-grid method

is not obtained. To predict the practical factor, a smoothing factor that is a contraction number of the smoothing method with respect to rough components is used.

Exploiting discrete Fourier components, smoothing factors of damped Jacobi, GS, ILU [59] and RB-GS [61] has been obtained.

5.4.1 Discrete Fourier sine transform

For simplicity, consider only in one dimension, however, extension to multidimension is straightforward. Let $I = \{1,\ 2,\ \ldots,\ n-1\}$. Every grid function $\boldsymbol{u} \mid I \rightarrow \Re$ is written as

$$\boldsymbol{u} = \sum_{k=1}^{n-1} c_k \boldsymbol{\psi}(\theta_k), \tag{5.37}$$

where

$$\boldsymbol{\psi}(\theta_k) = (\sin\theta_k,\ \sin 2\theta_k,\ \ldots,\ \sin(n-1)\theta_k)^T, \quad \theta_k = \frac{\pi k}{n} \tag{5.38}$$

with

$$c_k = \frac{2}{n}\boldsymbol{u}^T\boldsymbol{\psi}(\theta_k). \tag{5.39}$$

This is shown by the orthogonality

$$\boldsymbol{\psi}(\theta_k)^T\boldsymbol{\psi}(\theta_l) = \begin{cases} \dfrac{n}{2}, & \text{if } k = l \\ 0, & \text{otherwise} \end{cases} \tag{5.40}$$

5.4.2 Fourier smoothing factor

Let a smoothing iteration matrix be S, the error $\boldsymbol{e}_\nu$ after ν iterations is

$$\boldsymbol{e}_\nu = S^\nu \boldsymbol{e}_0, \tag{5.41}$$

with $\boldsymbol{e}_0$ an initial error. If the operator S has a complete set of eigenfunctions or *local modes* denoted by $\boldsymbol{\psi}(\theta)$, $\theta \in \Theta$, with Θ some discrete index set. Hence,

$$S^\nu\boldsymbol{\psi}(\theta) = \lambda^\nu(\theta)\boldsymbol{\psi}(\theta) \tag{5.42}$$

with $\lambda(\theta)$ the eigenvalue belonging to $\boldsymbol{\psi}(\theta)$. The eigenvalue $\lambda(\theta)$ is also called the *amplification factor* of the local mode $\boldsymbol{\psi}(\theta)$.

Since on the coarse grid the rapidly varying function cannot be approximated, there is no hope that the part of the error can be approximated on the coarse grid. The eigenfunctions $\boldsymbol{\psi}(\theta)$ are distinguished between *smooth* eigenfunctions ($\theta \in \Theta_s$) and *rough* eigenfunctions ($\theta \in \Theta_r$):

$$\Theta = \Theta_s \cup \Theta_r, \quad \Theta_s \cap \Theta_r = \emptyset. \tag{5.43}$$

In the case of the Fourier sine series, they are defined by

$$\Theta = \{\theta \mid \frac{\pi k}{n_1},\ k = 1,\ 2,\ \ldots,\ n_1 - 1\}, \quad \Theta_r = \Theta \cap [\frac{\pi}{2},\ \pi], \quad \Theta_s = \Theta \setminus \Theta_r. \tag{5.44}$$

Definition 5.4.1 (Local mode smoothing factor) The local mode smoothing factor ρ of the smoothing method is defined by

$$\rho = \sup\{|\lambda(\theta)| \mid \theta \in \Theta_r\}. \tag{5.45}$$

Hence, after ν smoothings the amplitude of the rough components of the error are multiplied by a factor ρ^ν or smaller. Note that the local mode smoothing factor is defined by the amplification factor in only rough part of the error. The local mode smoothing factor depends on the mesh size because Θ_r depends on the mesh size. In order to obtain a mesh-independent condition, we define a set $\bar{\Theta}_r = \cup\Theta_r$ with $\bar{\Theta}_r$ independent of the mesh size, and define

$$\bar{\rho} = \sup\{|\lambda(\theta)| \mid \theta \in \bar{\Theta}_r\} \tag{5.46}$$

so that

$$\rho \leq \bar{\rho}. \tag{5.47}$$

In the case of the Fourier sine series, $\bar{\Theta}_r$ are defined by

$$\bar{\Theta}_r = \left[\frac{\pi}{2},\ \pi\right). \tag{5.48}$$

This type of Fourier analysis and definition Eq. (5.46) of the smoothing factor have been introduced by Brandt [9].

Chapter 6

MULTIGRID PRECONDITIONED CONJUGATE GRADIENT METHOD

This chapter gives a sufficient condition for the MG method that can be used as a preconditioner of the CG method, and formulates the multigrid preconditioned conjugate gradient (MGCG) method [52, 32]. Let us start with a two-grid preconditioner in Section 6.1, and investigate a sufficient condition for the MG method in Section 6.2. In Section 6.3, the MGCG method is formulated. Historical remarks are described in Section 6.4.

6.1 Two-grid Preconditioner

This section investigates two sufficient conditions of the two-grid method for a preconditioner of the CG method. One is the two-grid method with pre-smoothing only ($\nu_1 \neq 0,\ \nu_2 = 0$) and the other is that with both smoothings ($\nu_1 \neq 0,\ \nu_2 \neq 0$).

Consider an $n \times n$ linear equation $L_l \boldsymbol{x}_l = \boldsymbol{f}_l$. As described in Section 3.1, the approximation after ν smoothing iterations is given by

$$\boldsymbol{u} = H^{\nu}\boldsymbol{u} + R\boldsymbol{f}, \tag{6.1}$$

where $H = P^{-1}Q$, $R = \sum_{i=0}^{\nu-1} H^i\ P^{-1}$ and $L_l = P - Q$. One two-grid iteration is represented by Figure 6.1 with $\boldsymbol{u}$ an initial approximation. r and p denote a restriction matrix and a prolongation matrix respectively.

6.1.1 Two-grid preconditioner with pre-smoothing only

From Figure 6.1, for $\nu_2 = 0$ the two-grid preconditioning matrix M is represented by

$$M^{-1} = R + pL_{l-1}^{-1}r(I - L_l R). \tag{6.2}$$

$\boldsymbol{u} = H^{\nu_1}\boldsymbol{u} + R\boldsymbol{f}$ // pre-smoothing
$\boldsymbol{d} = r(L_l\boldsymbol{u} - \boldsymbol{f})$ // coarse-grid correction
$\boldsymbol{v} = L_{l-1}^{-1}\boldsymbol{d}$
$\boldsymbol{u} = \boldsymbol{u} - p\boldsymbol{v}$
$\boldsymbol{u} = H^{\nu_2}\boldsymbol{u} + R\boldsymbol{f}$ // post-smoothing

Figure 6.1: The two-grid iteration

This means the approximation $\boldsymbol{u}$ after one two-grid iteration for $L_l\boldsymbol{x} = \boldsymbol{f}$ with zero initial approximation is represented by

$$\boldsymbol{u} = M^{-1}\boldsymbol{f}. \tag{6.3}$$

As described in Section 4.3, to satisfy the condition of a preconditioner of the CG method, M should be s.p.d.. The following theorem is useful to show the condition of the two-grid method.

Theorem 6.1.1 *Let $A = P - Q$ be symmetric and positive definite with P symmetric and nonsingular. R is symmetric and for any $\nu \geq 1$*

1. *If ν is odd, R is positive definite if and only if P is positive definite.*
2. *If ν is even, R is positive definite if and only if $P + Q$ is positive definite.*

Proof. See Ortega [42]. □

Let $\nu = \nu_1$, the following theorem holds.

Theorem 6.1.2 (Tatebe 1993) *Let L_{l-1}^{-1} be symmetric and positive definite, $r = p^*$ and $N = I - L_lR = H^\nu$. If H and P be symmetric and N be nonsingular, M is symmetric with respect to the N-energy inner product.*

1. *If ν is odd and P and H are positive definite, M is positive definite with respect to the N-energy inner product,*
2. *If ν is even and $P+Q$ is positive definite, M is positive definite with respect to the N-energy inner product,*

provided that N-energy inner product $(\boldsymbol{x},\ \boldsymbol{y})_N = (\boldsymbol{x},\ N\boldsymbol{y})$.

Proof. Since $N = I - L_lR = H^\nu$,

$$(\boldsymbol{x},\ M^{-1}\boldsymbol{y})_N = \boldsymbol{x}^T H^\nu R\boldsymbol{y} + \boldsymbol{x}^T H^\nu p L_{l-1}^{-1} r H^\nu \boldsymbol{y}. \tag{6.4}$$

Since P is symmetric, R is also symmetric, and since $(pL_{l-1}^{-1}rH^{\nu})^T = H^{\nu}pL_{l-1}^{-1}r$,

$$(M^{-1}\boldsymbol{x},\ \boldsymbol{y})_N = \boldsymbol{x}^T RH^{\nu}\boldsymbol{y} + \boldsymbol{x}^T H^{\nu}pL_{l-1}^{-1}rH^{\nu}\boldsymbol{y}. \tag{6.5}$$

Because $H = P^{-1}Q$ is symmetric,

$$\begin{aligned} H^{\nu}R &= (P^{-1}Q)^{\nu}\sum_{i=0}^{\nu-1}(P^{-1}Q)^i P^{-1} \\ &= \sum_{i=0}^{\nu-1} P^{-1}(QP^{-1})^i(QP^{-1})^{\nu} = RH^{\nu}, \end{aligned} \tag{6.6}$$

therefore M^{-1} is symmetric with respect to N-energy inner product.

Next, since $N = H^{\nu}$,

$$\begin{aligned} NM^{-1} &= (I - L_l R)\{R + pL_{l-1}^{-1}r(I - L_l R)\} \\ &= H^{\nu}R + H^{\nu}pL_{l-1}^{-1}rH^{\nu}. \end{aligned} \tag{6.7}$$

Assume ν is odd. From assumption, P is positive definite, thus R is s.p.d. by Theorem 6.1.1. Because H is s.p.d., H^{ν} is also s.p.d., thus $H^{\nu}R$ is positive definite (see [42]). Next, assume ν is even. Since H is symmetric, H has real eigenvalues, so that H^{ν} is positive definite. Since $P + Q$ is positive definite, then R is positive definite by Theorem 6.1.1, thus $H^{\nu}R$ is positive definite.

Since H^{ν} is symmetric and $pL_{l-1}^{-1}r$ is semi-positive definite, $H^{\nu}pL_{l-1}^{-1}rH^{\nu}$ is semi-positive definite. Therefore NM^{-1} is positive definite, so that M^{-1} is positive definite with respect to N-energy inner product. Thus, M is symmetric and positive definite w.r.t. N-energy inner product. □

A smoothing method that satisfies the assumption of Theorem 6.1.2; H is symmetric and nonsingular and P is symmetric, is the damped Richardson method. Thus if Q is positive definite, the two-grid preconditioner with the damped Richardson pre-smoother is a valid preconditioner of the CG method which uses the N-energy inner product instead of the usual inner product.

6.1.2 Two-grid preconditioner with both pre- and post-smoothings

Next consider the two-grid preconditioner with both pre- and post-smoothings. Let the iteration matrices of pre- and post-smoothers be $H_1 = P_1^{-1}Q_1$ and $H_2 = P_2^{-1}Q_2$ respectively, the two-grid preconditioning matrix M is represented by

$$\begin{aligned} M^{-1} &= H_2^{\nu_2}\left\{(I - pL_{l-1}^{-1}rL_l)R_1 + pL_{l-1}^{-1}r\right\} + R_2 \\ &= H_2^{\nu_2}R_1 + R_2 + H_2^{\nu_2}pL_{l-1}^{-1}r(I - L_l R_1). \end{aligned} \tag{6.8}$$

Since $R_1 = \sum_{i=0}^{\nu_1-1}(P_1^{-1}Q_1)^i P_1^{-1}$,

$$I - L_l R_1 = (Q_1 P_1^{-1})^{\nu_1}. \tag{6.9}$$

Therefore Eq. (6.8) is rewritten as

$$M^{-1} = H_2^{\nu_2} R_1 + R_2 + H_2^{\nu_2} p L_{l-1}^{-1} r (Q_1 P_1^{-1})^{\nu_1}. \tag{6.10}$$

Theorem 6.1.3 (Tatebe 1993) *Let L_{l-1}^{-1} be symmetric and positive definite and $r = p^*$. If $P_1 = P_2 = P_1^T$, $\rho(H_1) < 1$ and $\nu_1 = \nu_2 = \nu$, M is symmetric and positive definite.*

Proof. Let $P = P_1 = P_2$ and $H = H_1 = H_2$, we have

$$M^{-1} = H^\nu R + R + H^\nu p L_{l-1}^{-1} r (H^T)^\nu, \tag{6.11}$$

since $Q_1 P_1^{-1} = H^T$. Because $R = \sum_{i=0}^{\nu-1} H^i \ P^{-1}$,

$$H^\nu R + R = H^\nu \sum_{i=0}^{\nu-1} H^i \ P^{-1} + \sum_{i=0}^{\nu-1} H^i \ P^{-1} = \sum_{i=0}^{2\nu-1} H^i \ P^{-1}, \tag{6.12}$$

thus $H^\nu R + R$ is the inverse of the preconditioning matrix of 2ν smoothing iterations. From Theorem 6.1.1, it is symmetric, so that M^{-1} is symmetric.

By Theorem 3.3.4, if P is s.p.d. and $\rho(H) < 1$, $P + Q$ are positive definite, thus $H^\nu R + R$ is positive definite. Since L_{l-1}^{-1} is positive definite, $H^\nu p L_{l-1}^{-1} r (H^T)^\nu$ is semi-positive definite. Therefore M^{-1} is positive definite, so that M is symmetric and positive definite. □

Smoothing methods which satisfy the assumption of Theorem 6.1.3; P is symmetric, are the damped Jacobi method, the Red-Black symmetric Gauss-Seidel (RB-SGS) method, the multicolor SSOR method, the symmetric ADI method, the CG method and so on. From this theorem, the two-grid preconditioner with one of these iterative methods as pre- and post-smoothers fulfills the condition of preconditioner of the CG method.

For $m_1 = m_2 = 1$, the following theorem holds [32, 33].

Theorem 6.1.4 (Kettler 1981) *Let L_{l-1}^{-1} be symmetric and positive definite and $r = p^*$. If $P_1 = P_2^T$, $\rho(P_1^{-1} Q_1) < 1$ and $\nu_1 = \nu_2 = 1$, M is symmetric and positive definite.*

Although I could not get their proof, it is proved as follows.

Proof. The two-grid preconditioning matrix for $\nu_1 = \nu_2 = 1$ is

$$M^{-1} = P_2^{-1} Q_2 P_1^{-1} + P_2^{-1} + P_2^{-1} Q_2 p L_{l-1}^{-1} r Q_1 P_1^{-1}. \tag{6.13}$$

Let $P = P_1 = P_2^T$,

$$M^{-1} = P^{-T} Q^T P^{-1} + P^{-T} + P^{-T} Q^T p L_{l-1}^{-1} r Q P^{-1}. \tag{6.14}$$

Because $Q^T = P^T - L_l$,

$$P^{-T} Q^T P^{-1} + P^{-T} = P^{-T} (P^T - L_l) P^{-1} + P^{-T} = P^{-T} L_l P^{-1} + P^{-1} + P^{-T}. \tag{6.15}$$

Thus $P^{-T}Q^TP^{-1} + P^{-T}$ is symmetric. Moreover, Eq. (6.15) follows

$$P^{-T}Q^TP^{-1} + P^{-T} = P^{-T}(P + Q^T)P^{-1}. \tag{6.16}$$

Hence, this matrix is positive definite if $P + Q^T$ is positive definite. However, since $\rho(P^{-1}Q) < 1$, $P+Q$ is positive definite, thus $P+Q^T$ is positive definite. Therefore $P^{-T}Q^TP^{-1}+P^{-T}$ is positive definite. Thus $P_2^{-1}Q_2pL_{l-1}^{-1}rQ_1P_1^{-1}$ is symmetric and semi-positive definite, so that M^{-1} and M are symmetric and positive definite. □

In fact, Theorem 6.1.4 holds for not only $\nu_1 = \nu_2 = 1$ but also $\nu_1 = \nu_2 \geq 1$.

Theorem 6.1.5 *Let L_{l-1}^{-1} be symmetric and positive definite and $r = p^*$. If $P_1 = P_2^T$, $\rho(P_1^{-1}Q_1) < 1$ and $\nu_1 = \nu_2$, M is symmetric and positive definite.*

Proof. Let $P = P_1 = P_2^T$ and $\nu = \nu_1 = \nu_2$, after ν pre-smoothing iterations,

$$\boldsymbol{u} = (P^{-1}Q)^\nu \boldsymbol{u} + \sum_{i=0}^{\nu-1}(P^{-1}Q)^iP^{-1}\boldsymbol{f}, \tag{6.17}$$

similarly, after ν post-smoothing iterations,

$$\boldsymbol{u} = (P^{-T}Q^T)^\nu \boldsymbol{u} + \sum_{i=0}^{\nu-1}(P^{-T}Q^T)^iP^{-T}\boldsymbol{f}. \tag{6.18}$$

Hence, the preconditioning matrix of ν post-smoothing iterations is adjoint of that of ν pre-smoothing iterations. Furthermore, since $\rho(P_1^{-1}Q_1) < 1$,

$$\rho\left((P_1^{-1}Q_1)^\nu\right) = \rho^\nu(P_1^{-1}Q_1) < \rho(P_1^{-1}Q_1) < 1, \tag{6.19}$$

thus the proof is complete using Theorem 6.1.4. □

From Theorem 6.1.5, smoothers are not necessary to be symmetric, and furthermore, their number of iterations is not necessary to be one. For example, we can use 2 RB-GS iterations for pre-smoothing and 2 BR-GS iterations for post-smoothing.

6.2 Multigrid Preconditioner

In the previous section, two sufficient conditions of the two-grid preconditioner are considered. In this section, they are extended for the MG preconditioner. In the V-cycle MG method, the following theorem holds.

Theorem 6.2.1 *If the assumption of Theorem 6.1.2 or 6.1.5 is satisfied on each grid, V-cycle MG preconditioning matrix is symmetric and positive definite.*

Proof. When the assumption of Theorem 6.1.5 is satisfied, the V-cycle MG preconditioning matrix M_l with l grid levels is recursively defined by

$$\begin{aligned} M_0^{-1} &= L_0^{-1} \text{ or } R \\ M_i^{-1} &= H_2^m R_1 + R_2 + H_2^m p M_{i-1}^{-1} r (H_2^T)^m \quad (i \geq 1) \end{aligned} \tag{6.20}$$

M_0^{-1} is symmetric and positive definite. If M_{i-1}^{-1} is symmetric and positive definite, M_i^{-1} is also symmetric and positive definite because of Theorem 6.1.5. By mathematical induction, every M_i^{-1} $(i \geq 0)$ is symmetric and positive definite, so that every M_i is s.p.d.. Therefore the V-cycle MG method satisfies the condition of the preconditioner of the CG method. For the assumption of Theorem 6.1.2, the theorem is shown by the similar way. □

Note that Theorem 6.2.1 requires only a convergent smoother on each grid, hence the smoothing property and the approximation property are not always necessary for preconditioning. Furthermore, the V-cycle MG method is not necessary to be convergent.

For W-cycle and other MG cycles, the following theorem holds.

Theorem 6.2.2 *Let the assumption of Theorem 6.1.2 or 6.1.5 be satisfied on each grid. If the MG method called on each grid is convergent, all MG$(n,\ m)$ methods satisfy conditions of the preconditioner of the CG method, where n is a MG cycle and m is the number of smoothing iterations.*

Proof. When the assumption of Theorem 6.1.5 is satisfied, the MG preconditioning matrix $M_l^{(n)}$ with n MG calls as the approximate solution on the coarse grid is defined by

$$\begin{aligned} \left(M_0^{(n)}\right)^{-1} &= L_0^{-1} \text{ or } R \\ \left(M_i^{(n)}\right)^{-1} &= \sum_{i=0}^{n-1} H_{\mathrm{mg}}^i \left\{ H_2^m R_1 + R_2 + H_2^m p \left(M_{i-1}^{(n)}\right)^{-1} r \left(H_2^T\right)^m \right\}, \quad (i \geq 1) \end{aligned} \tag{6.21}$$

where $H_{\mathrm{mg}} = H_2^m \left\{ I - p \left(M_{i-1}^{(n)}\right)^{-1} r L_i \right\} H_1^m$. $\left(M_0^{(n)}\right)^{-1}$ is symmetric and positive definite. If $\left(M_{i-1}^{(n)}\right)^{-1}$ is symmetric and positive definite, $H_2^m R_1 + R_2 + H_2^m p \left(M_{i-1}^{(n)}\right)^{-1} r \left(H_2^T\right)^m$ is symmetric and positive definite by Theorem 6.1.5, so that $\left(M_i^{(n)}\right)^{-1}$ is symmetric.

From the assumption, $\rho(H_{\mathrm{mg}}) < 1$, so that $\left(M_i^{(n)}\right)^{-1}$ is positive definite. Thus, $M_i^{(n)}$ is symmetric and positive definite. The W-cycle MG method is a special case with $n = 2$. Therefore the W-cycle MG method and all MG$(n,\ m)$ $(n,\ m \geq 1)$ satisfy the condition of the preconditioner of the CG method. It is shown by the similar way when the assumption of Theorem 6.1.2 is satisfied. □

In the W-cycle MG preconditioner, the assumption $\rho(H_{\mathrm{mg}}) < 1$ on each grid, which means each (V-cycle) MG iteration is convergent, is required since the preconditioning matrix should be

positive definite. Theorem 6.2.2 ensures not only MG(n, m) method but also the MG method which calls any number of identical MG calls for the solution on the coarse grid, for example, we can use two MG calls on the grid level 3 and one MG call on the grid level 2. However, the F-cycle MG method is not a valid preconditioner for the CG method.

6.3 MGCG Method

In the previous sections, we have two sufficient conditions for the MG preconditioner; one is for the MG method with pre-smoothing only and the other is for that with both smoothings. With only pre-smoothing, the MG preconditioner with the damped Richardson smoother is a valid preconditioner of the CG method with the N-energy inner product instead of the usual inner product. If N is explicitly computed and stored to the memory, computational work of N-energy inner product is quite expensive, since N is dense. Still if not, computational cost of N-energy inner product is nearly same as that of the pre-smoothing method.

On the other hand, with both pre- and post-smoothings, the V-cycle MG method is a valid preconditioner of the CG method if

$$r = p^*, \quad \mu_1 = \mu_2 \neq 0, \tag{6.22}$$

and if post-smoother is convergent and adjoint of pre-smoother. Of course, pre- and post-smoothers may be identical and symmetric.

We denote MGCG(l, m, n, g) for the MGCG method with n iterations of l smoother and g grid level. m means a cycle of the MG preconditioner. l smoother is used as pre-smoother and its adjoint is used as post-smoother. When g is omitted, the MGCG method uses all available grid levels. For example, MGCG(RB-SGS, 1, 2) uses the V-cycle MG preconditioner with two iterations of the symmetric RB-GS pre- and post-smoothers, and MGCG(RB-GS, 2, 1) uses the W-cycle MG preconditioner with one RB-GS pre-smoothing and one BR-GS post-smoothing.

6.4 Historical Remark

In 1981, Kettler and Meijerink have proposed the MGCG method in [32, 33]. As described in Section 6.1, they used Theorem 6.1.4 for a sufficient condition of the MG preconditioner. In 1991, Wesseling wrote a sufficient condition of the MG preconditioner in his book [59], which is same as Theorem 6.1.3, however the explanation of the condition is incorrect. Preconditioner of the CG method should have a symmetric and positive definite preconditioning matrix not a symmetric iteration matrix.

Kettler and Meijerink performed numerical experiments in [32] on Stone's test problem [50], which is the anisotropic diffusion equation, and Kershaw's test problem [31], which is a Helmholtz equation. They used line Gauss-Seidel smoother, ILU smoother, alternative line Gauss-Seidel smoother and incomplete line LU smoother as the smoother of the MG preconditioner. They concluded that MG, MGCG and ICCG are equally efficient for Stone's problem if the smoother of MG and MGCG is ILLU. For Kershaw's problem, MG is significantly slower than MGCG and ICCG. ICCG suffers from a slow start and is rather inefficient for Poisson-type equation. Hence MGCG is to be preferred.

Chapter 7

RATE OF CONVERGENCE OF THE MGCG METHOD

It is proved that the MGCG method is robust for the Poisson Equation. Fast rate of convergence of the MGCG method is shown by the eigenvalue analysis of the preconditioned matrix. Exploiting Fourier components on each grid level, all the eigenvalues of the MG preconditioned matrix are analytically obtained. Specifically, the eigenvalues of the two-grid preconditioned matrix are investigated with damped Jacobi smoother or Red-Black Gauss-Seidel (RB-GS) smoother for 1- and 2-dimensional Poisson equations. For 1 dimension, it is shown the MG preconditioner with RB-GS smoother is an exact solver. For 2 dimensions, the eigenvalues of the two-grid preconditioned matrix lie in $\frac{3}{4} \leq \lambda \leq 1$.

7.1 Two-grid Preconditioner

We start with the simplest form of the MG preconditioner, which is the two-grid preconditioner that exploits only two grid levels. Consider a linear equation $L_2 \boldsymbol{u}^{(2)} = \boldsymbol{f}^{(2)}$ on the grid Ω_2, which is the finest grid, such that L_2 is symmetric and positive definite. Superscript or subscript that denotes the grid level may be omitted when it is clear. Let an s.p.d. coefficient matrix on a coarse grid Ω_1 be L_1, the prolongation and restriction operators be p and r, respectively. Let iteration matrices of pre- and post-smoothers be $P_1^{-1}Q_1$ and $P_2^{-1}Q_2$, respectively, and the number of iterations be m_1 and m_2. The two-grid preconditioner is shown by Tab. 7.1. In the following investigation, L_1 is constructed by discretization coarse-grid approximation. To satisfy a condition of preconditioner of the CG method, r is adjoint of p, namely, $p = r^*$, and $P_2 = P_1^T = P^T$ and

Table 7.1: The two-grid iteration

$\boldsymbol{u} = (P_1^{-1}Q_1)^{m_1}\boldsymbol{u} + \sum_{i=0}^{m_1-1}(P_1^{-1}Q_1)^i P_1^{-1}\boldsymbol{f}$	// pre-smoothing
$\boldsymbol{u} = \boldsymbol{u} - p\, L_{l-1}^{-1} r\,(L_l\boldsymbol{u} - \boldsymbol{f})$	// coarse-grid correction
$\boldsymbol{u} = (P_2^{-1}Q_2)^{m_2}\boldsymbol{u} + \sum_{i=0}^{m_2-1}(P_2^{-1}Q_2)^i P_2^{-1}\boldsymbol{f}$	// post-smoothing

$m_1 = m_2 = m(\geq 1)$. We obtain the two-grid preconditioned matrix

$$B = (P^{-T}Q^T)^m \sum_{i=0}^{m-1}(P^{-1}Q)^i P^{-1} L_2 + \sum_{i=0}^{m-1}(P^{-T}Q^T)^i P^{-T} L_2 + (P^{-T}Q^T)^m p L_1^{-1} r (QP^{-1})^m L_2. \tag{7.1}$$

When the smoothing method is symmetric,

$$B = \sum_{i=0}^{2m-1}(P^{-1}Q)^i P^{-1} L_2 + (P^{-1}Q)^m p L_1^{-1} r (QP^{-1})^m L_2. \tag{7.2}$$

7.2 One-dimensional Poisson Equation

Consider the following one-dimensional Poisson equation:

$$-\frac{d^2u(x)}{dx^2} = f(x) \quad \text{with} \quad u(0) = 0,\ \frac{du(1)}{dx} = 0. \tag{7.3}$$

Note that the following analysis of Neumann-Dirichlet boundary condition can be easily extended to that of Dirichlet-Dirichlet boundary condition. Discretizing the range $(0, 1)$ into N sections, Eq. (7.3) can be translated to a system of N equations,

$$\frac{1}{h^2}\begin{pmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & & \ddots & & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 1 \end{pmatrix}\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{N-1} \\ u_N \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{N-1} \\ f_N \end{pmatrix}, \tag{7.4}$$

where $h = \frac{1}{N}$, $u_i = u(\frac{i}{N})$ and $f_i = f(\frac{i}{N})$. For simplicity, assume N is even.

7.2.1 Damped Jacobi smoother

Let a matrix $P = \frac{1}{\omega}\mathrm{diag}(L_2)$, a damped Jacobi smoother iterates

$$P\boldsymbol{x}^{i+1} = Q\boldsymbol{x}^i + \boldsymbol{f}, \tag{7.5}$$

where $Q = P - L_2$, with $\boldsymbol{x}^0$ an initial approximation. Without difficulty, the eigenvalues of the iteration matrix $P^{-1}Q$ are $1-\omega+\omega\cos\alpha_i$ where $\alpha_i^{(2)}=(\frac{1}{2}+i)\frac{\pi}{N_2}$ $(i = 0,\ 1,\ 2,\ \cdots,\ N_2-1)$, and the corresponding eigenvectors are

$$\boldsymbol{\nu}_i^{(2)} = (\sin\alpha_i^{(2)}\ \ \sin 2\alpha_i^{(2)}\ \ \cdots\ \ \sin N\alpha_i^{(2)})^T.$$

Note that this also means that the damped Jacobi method is convergent if and only if $0 < \omega \leq 1$. For the Dirichlet boundary condition, let $\alpha_i^{(2)} = (1+i)\frac{\pi}{N_2}$ $(i = 0,\ 1,\ 2,\ \cdots,\ N_2-2)$, and replacing $N-i-1$ to $N-i-2$, the following discussion is applicable.

Theorem 7.2.1 *The eigenvalues of the two-grid preconditioned matrix with m iterations of damped Jacobi smoother are 1 and*

$$1 - \frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m} - \frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^{2m}$$

on the 1-dimensional Poisson equation.

Proof. Since $P^{-1}L_l = P^{-1}(P-Q) = I - P^{-1}Q$, the eigenvalues and the corresponding eigenvectors of $P^{-1}L_l$ are $\omega(1-\cos\alpha_i)$ and $\boldsymbol{\nu}_i$, respectively. Thus, the eigenvalues of the first term of Eq. (7.2) are

$$\sum_{i=0}^{2m-1}(1-\omega+\omega\cos\alpha_i)^i\omega(1-\cos\alpha_i) = 1-(1-\omega+\omega\cos\alpha_i)^{2m}. \tag{7.6}$$

In the one-dimensional case, the stencils of r and p are

$$[r]_k = \frac{1}{4}[1\ 2\ 1] \quad \text{and} \quad [p^T]_k = \frac{1}{2}[1\ 2\ 1], \tag{7.7}$$

where $k \in \Omega_1$, respectively. Thus,

$$r\boldsymbol{\nu}_i^{(2)} = \frac{1+\cos\alpha_i^{(2)}}{2}\boldsymbol{\nu}_i^{(1)}, \tag{7.8}$$

where $\boldsymbol{\nu}_i^{(1)} = (\sin\alpha_i^{(1)}\ \ \sin 2\alpha_i^{(1)}\ \ \cdots\ \ \sin N_1\alpha_i^{(1)})^T$, $\alpha_i^{(1)} = (\frac{1}{2}+i)\frac{\pi}{N_1}$ $(i = 0,\ 1,\ 2,\ \cdots,\ N_1-1)$ and $N_1 = \frac{N_2}{2}$ on the grid level 1. Note that $\boldsymbol{\nu}_{N-i-1}^{(1)} = -\boldsymbol{\nu}_i^{(1)}$. Thus,

$$P_1^{-1}rQ_2\boldsymbol{\nu}_i^{(2)} = 2(1-\omega+\omega\cos\alpha_i)(1+\cos\alpha_i)\boldsymbol{\nu}_i^{(1)} \tag{7.9}$$

Moreover, since $\alpha_{N_2-i-1}^{(2)} = -\alpha_i^{(2)} + \pi$,

$$\sin j\alpha_{N-i-1}^{(2)} = \begin{cases} \sin j\alpha_i^{(2)}, & \text{if } j \text{ is odd} \\ -\sin j\alpha_i^{(2)}, & \text{otherwise.} \end{cases} \tag{7.10}$$

Thus, $p\boldsymbol{\nu}_i^{(1)}$ is represented using $\boldsymbol{\nu}_i^{(2)}$ and $\boldsymbol{\nu}_{N-i-1}^{(2)}$,

$$\begin{aligned} p\boldsymbol{\nu}_i^{(1)} &= \frac{\cos\alpha_i^{(2)}}{2}(\boldsymbol{\nu}_i^{(2)} + \boldsymbol{\nu}_{N-i-1}^{(2)}) + \frac{1}{2}(\boldsymbol{\nu}_i^{(2)} - \boldsymbol{\nu}_{N-i-1}^{(2)}) \\ &= \frac{1+\cos\alpha_i}{2}\boldsymbol{\nu}_i^{(2)} - \frac{1-\cos\alpha_i}{2}\boldsymbol{\nu}_{N-i-1}^{(2)}. \end{aligned} \tag{7.11}$$

On the other hand, the second term of Eq. (7.2) is rewritten as

$$(P^{-1}Q)^m p L_1^{-1} r (QP^{-1})^m L_2 = (P_2^{-1}Q_2)^m p (P_1^{-1}L_1)^{-1} P_1^{-1} r Q_2 (P_2^{-1}Q_2)^{m-1} P_2^{-1} L_2.$$

Subscripts denote the grid level. Therefore,

$$\begin{aligned} B\boldsymbol{\nu}_i &= \left(1 - \frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m}\right)\boldsymbol{\nu}_i^{(2)} \\ &\quad -\frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^m(1-\omega-\omega\cos\alpha_i)^m\boldsymbol{\nu}_{N-i-1}^{(2)} \end{aligned} \tag{7.12}$$

Similarly, since $\cos\alpha_{N-i-1} = -\cos\alpha_i$,

$$\begin{aligned} B\boldsymbol{\nu}_{N-i-1} &= -\frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^m(1-\omega+\omega\cos\alpha_i)^m\boldsymbol{\nu}_i^{(2)} \\ &\quad + \left(1 - \frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^{2m}\right)\boldsymbol{\nu}_{N-i-1}^{(2)}, \end{aligned} \tag{7.13}$$

thus, there are two eigenvectors of B in Span$\{\boldsymbol{\nu}_i,\ \boldsymbol{\nu}_{N-i-1}\}$. Let $k_1\boldsymbol{\nu}_i + k_2\boldsymbol{\nu}_{N-i-1}$ be the eigenvectors and λ be the corresponding eigenvalues,

$$B(k_1\boldsymbol{\nu}_i + k_2\boldsymbol{\nu}_{N-i-1}) = \lambda(k_1\boldsymbol{\nu}_i + k_2\boldsymbol{\nu}_{N-i-1}). \tag{7.14}$$

From Eq. (7.12), (7.13) and (7.14), the following equation

$$\begin{aligned} \lambda^2 - \left(2 - \frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m} - \frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^{2m}\right)\lambda \\ +1 - \frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m} - \frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^{2m} &= 0 \end{aligned} \tag{7.15}$$

holds. Thus,

$$\lambda = 1,\ 1 - \frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m} - \frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^{2m} \tag{7.16}$$

□

Therefore, half of eigenvalues are unity. Note that the two-grid method with $\omega = 1$ is not robust, since the minimum eigenvalue $(1 - \cos^{2m}\alpha_0)$ approaches to 0 as $N \to \infty$. For $m = 1$, the next lemma is shown.

Lemma 7.2.2 *For $m = 1$, optimal condition number of the preconditioned matrix is $\frac{9}{8}$ when $\omega = \frac{2}{3}$.*

Proof. From Eq. (7.16), the maximum eigenvalue is 1. Thus, the condition number is minimum when the minimum eigenvalue is maximum.

$$\begin{aligned}
&\max_{\omega}\min_{\alpha_i}\ 1-\frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^2-\frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^2\\
&=\quad \max_{\omega}\min_{\alpha_i}\ 1-(1-\omega)^2-\omega(3\omega-2)\cos^2\alpha_i\\
&=\quad \frac{8}{9},
\end{aligned}$$

when $\omega=\frac{2}{3}$. □

Corollary 7.2.3 *For $m=1$ and $\omega=\frac{2}{3}$, the MGCG method with two grid levels converges at two iterations.*

Proof. For $\omega=\frac{2}{3}$, $\lambda=\frac{8}{9},\ 1$. □

Note that the optimal smoothing factor is $\frac{1}{9}$ when $\omega=\frac{2}{3}$, which is equivalent to the maximum eigenvalue of the two-grid iteration matrix obtained here.

For the Dirichlet boundary condition, the following theorem holds:

Theorem 7.2.4 *For one-dimensional Poisson equation with the Dirichlet boundary condition, the eigenvalues of the two-grid preconditioned matrix with m damped Jacobi iterations are unity, $1-(1-\omega)^{2m}$ and*

$$1-\frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m}-\frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^{2m},$$

where $\alpha_i=(i+1)\frac{\pi}{N}$, $i=0,\ 1,\ \ldots,\ \frac{N}{2}-2$.

Proof. Let

$$\boldsymbol{\nu}_i=(\sin\alpha_i\ \ \sin 2\alpha_i\ \ \cdots\ \ \sin(N-1)\alpha_i)^T, \tag{7.17}$$

for $i=0,\ 1,\ \ldots,\ \frac{N}{2}-2$, it is shown that eigenvalues are unity and

$$1-\frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m}-\frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^{2m}$$

by the similar way as the proof of Theorem 7.2.1. For $i=\frac{N}{2}-1$, then $\alpha_i=\frac{\pi}{2}$ and

$$r\boldsymbol{\nu}_i=0. \tag{7.18}$$

Hence

$$B\boldsymbol{\nu}_i=\left\{1-(1-\omega)^{2m}\right\}\boldsymbol{\nu}_i \tag{7.19}$$

Then the theorem holds. □

In this case, Corollary 7.2.3 also holds.

7.2.2 Red-Black Gauss-Seidel smoother

Let a matrix

$$P = \frac{1}{h^2}\begin{pmatrix} 2 & -1 & & & & \\ & 2 & & & & \\ & -1 & 2 & -1 & & \\ & & & \ddots & & \\ & & & -1 & 2 & -1 \\ & & & & & 1 \end{pmatrix}. \tag{7.20}$$

The RB-GS iterates

$$P\boldsymbol{x}^{i+1} = Q\boldsymbol{x}^i + \boldsymbol{f}, \tag{7.21}$$

where $Q = P - L$. Since P is not symmetric, it's adjoint smoother; the Black-Red Gauss-Seidel (BR-GS) iteration,

$$P^T\boldsymbol{x}^{i+1} = Q^T\boldsymbol{x}^i + \boldsymbol{f}, \tag{7.22}$$

is used as post-smoother.

Theorem 7.2.5 *The eigenvalues of the two-grid preconditioned matrix with the RB-GS smoother are unity.*

Proof. Let vectors $\boldsymbol{\nu}_i = (\sin\alpha_i \ \sin 2\alpha_i \ \cdots \ \sin N\alpha_i)^T$, the following equations hold:

$$P^{-1}Q\boldsymbol{\nu}_i = \frac{\cos\alpha_i(1+\cos\alpha_i)}{2}\boldsymbol{\nu}_i - \frac{\cos\alpha_i(1-\cos\alpha_i)}{2}\boldsymbol{\nu}_{N-i-1} \tag{7.23}$$

$$P^{-T}Q^T\boldsymbol{\nu}_i = \frac{\cos\alpha_i(1+\cos\alpha_i)}{2}\boldsymbol{\nu}_i + \frac{\cos\alpha_i(1-\cos\alpha_i)}{2}\boldsymbol{\nu}_{N-i-1} \tag{7.24}$$

And,

$$(P^{-1}L)^{-1}\boldsymbol{\nu}_i = \frac{2-\cos\alpha_i}{2(1-\cos\alpha_i)}\boldsymbol{\nu}_i - \frac{\cos\alpha_i}{2(1+\cos\alpha_i)}\boldsymbol{\nu}_{N-i-1} \tag{7.25}$$

Moreover,

$$P_{l-1}^{-1}rQ_l\boldsymbol{\nu}_i^{(2)} = \cos\alpha_i(1+2\cos^2\alpha_i)\boldsymbol{\nu}_i^{(1)} - \cos\alpha_i(1-2\cos^2\alpha_i)\boldsymbol{\nu}_{\frac{N}{2}-i-1}^{(1)} \tag{7.26}$$

Therefore,

$$B\boldsymbol{\nu}_i = \left(1 - \frac{\cos^3\alpha_i}{2}\right)\boldsymbol{\nu}_i - \frac{\cos^3\alpha_i(1-\cos\alpha_i)}{2(1+\cos\alpha_i)}\boldsymbol{\nu}_{N-i-1} \tag{7.27}$$

$$B\boldsymbol{\nu}_{N-i-1} = \frac{\cos^3\alpha_i(1+\cos\alpha_i)}{2(1-\cos\alpha_i)}\boldsymbol{\nu}_{N-i-1}\left(1 + \frac{\cos^3\alpha_i}{2}\right)\boldsymbol{\nu}_i \tag{7.28}$$

Now, let $k_1\boldsymbol{\nu}_i + k_2\boldsymbol{\nu}_{N-i-1}$ be the eigenvectors of the matrix B and λ be the corresponding eigenvalues. The following equation

$$\lambda^2 - 2\lambda + 1 = 0 \tag{7.29}$$

holds. Thus $\lambda = 1$. □

Corollary 7.2.6 *The MG preconditioner with the RB-GS smoother is an exact Poisson solver.*

Proof. By Theorem 7.2.5, the two-grid preconditioner is an exact solver. If the MG preconditioner with i grid levels is an exact solver, the MG preconditioner with $i+1$ grid levels is also an exact solver by Theorem 7.2.5. By mathematical induction, the corollary holds. □

Note that the black points updated first by the BR-GS post-smoother are included in $\Omega_2 \setminus \Omega_1$.

7.3 Two-dimensional Poisson Equation

In this section, we treat the two-dimensional Poisson equation:

$$-\left(\frac{\partial^2}{\partial x^2}+\frac{\partial^2}{\partial y^2}\right)u(x,y)=f(x,y) \quad \text{in} \quad \Omega=(0,1)\times(0,1)$$

$$\text{with} \quad \begin{cases} u=0 & \text{on } x=0 \text{ or } y=0 \\ \frac{\partial u}{\partial n}=0 & \text{on } x=1 \text{ or } y=1 \end{cases} \tag{7.30}$$

where $\frac{\partial}{\partial n}$ denotes the partial derivative with respect to outward direction of boundary. When this two-dimensional equation is discretized into $N \times N$ meshes, we get a block-tridiagonal matrix whose stencil is

$$\frac{1}{h^2}\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}, \tag{7.31}$$

where $h=\frac{1}{N}$. For simplicity, assume N is even.

7.3.1 Damped Jacobi smoother

Let $P=\frac{1}{\omega}\mathrm{diag}(L_l)$, eigenvectors of the damped Jacobi iteration matrix $P^{-1}Q$ are

$$\boldsymbol{\nu}_{ij}=(\sin\alpha_i \sin\beta_j \;\; \sin 2\alpha_i \sin\beta_j \;\; \cdots \;\; \sin N\alpha_i \sin N\beta_j)^T, \tag{7.32}$$

and the corresponding eigenvalues are $1-\omega+\frac{\omega(\cos\alpha_i+\cos\beta_j)}{2}$, where $\alpha_i=(\frac{1}{2}+i)\frac{\pi}{N}$ and $\beta_j=(\frac{1}{2}+j)\frac{\pi}{N}$ $(i,\ j=0,\ 1,\ \cdots,\ N-1)$. Thus, the eigenvalues of $P^{-1}L$ are $\omega\left(1-\frac{\cos\alpha_i+\cos\beta_j}{2}\right)$.

7.3.1.1 Semicoarsening

In semicoarsening, while mesh size of coarse grid is the same in at least one direction, that of coarse grid is double that of fine grid in other directions. In the semicoarsening case in the x-direction, the coefficient matrix on Ω_1 is

$$[L_1]_k=\frac{1}{4h^2}\begin{bmatrix} & -4 & \\ -1 & 10 & -1 \\ & -4 & \end{bmatrix}, \tag{7.33}$$

where $k \in \Omega_1$. Hence

$$[P_1^{-1}Q_1]_k = \omega \begin{bmatrix} & \frac{2}{5} & \\ \frac{1}{10} & \frac{1}{\omega} - 1 & \frac{1}{10} \\ & \frac{2}{5} & \end{bmatrix}. \tag{7.34}$$

Let

$$\boldsymbol{\nu}_{ij}^{(1)} = (\sin\alpha_i^{(1)} \sin\beta_j^{(1)} \quad \sin 2\alpha_i^{(1)} \sin\beta_j^{(1)} \quad \cdots \quad \sin(\frac{N}{2}-1)\alpha_i^{(1)} \sin(N-1)\beta_j^{(1)})^T, \tag{7.35}$$

eigenvectors of the iteration matrix $P_1^{-1}Q_1$ are $\boldsymbol{\nu}_{ij}^{(1)}$ and the corresponding eigenvalues are $1 - \omega + \frac{\omega(\cos\alpha_i^{(1)} + 4\cos\beta_j^{(1)})}{5}$, where $\alpha_i^{(1)} = (\frac{1}{2} + i)\frac{2\pi}{N} = 2\alpha_i^{(2)}$ and $\beta_j^{(1)} = \beta_j^{(2)}$. Thus, the eigenvalues of $P^{-1}L_{l-1}$ are $\omega\left(1 - \frac{\cos\alpha_i^{(1)} + 4\cos\beta_j^{(1)}}{5}\right)$.

The stencils of r and p are same in the one-dimensional case,

$$[r]_k = \frac{1}{4}[1\ 2\ 1] \quad \text{and} \quad [p^T]_k = \frac{1}{2}[1\ 2\ 1], \tag{7.36}$$

where $k \in \Omega_1$, respectively.

Theorem 7.3.1 *Let $A = 1-\omega+\frac{\omega}{2}\cos\beta_j$ and $B = \frac{\omega}{2}\cos\alpha_i$. The eigenvalues of the two-grid preconditioned matrix with damped Jacobi smoother and semicoarsening are equivalent to the eigenvalues of the following 2 × 2 matrices:*

$$C_{ij} = \begin{pmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{pmatrix}, \tag{7.37}$$

where $i = 0,\ 1,\ \cdots,\ \frac{N}{2}-1$, $j = 0,\ 1,\ \cdots,\ N-1$ and

$$\begin{aligned}
c_{00} &= 1 - (A+B)^{2m} + \frac{2}{5}\frac{\left(1 - \frac{\cos\alpha_i + \cos\beta_j}{2}\right)(1+\cos\alpha_i)^2}{1 - \frac{\cos 2\alpha_i + 4\cos\beta_j}{5}}(A+B)^{2m} \\
c_{01} &= -\frac{2}{5}\frac{\left(1 - \frac{\cos\alpha_i + \cos\beta_j}{2}\right)(1-\cos^2\alpha_i)}{1 - \frac{\cos 2\alpha_i + 4\cos\beta_j}{5}}(A+B)^m(A-B)^m \\
c_{10} &= -\frac{2}{5}\frac{\left(1 - \frac{-\cos\alpha_i + \cos\beta_j}{2}\right)(1-\cos^2\alpha_i)}{1 - \frac{\cos 2\alpha_i + 4\cos\beta_j}{5}}(A+B)^m(A-B)^m \\
c_{11} &= 1 - (A-B)^{2m} + \frac{2}{5}\frac{\left(1 - \frac{-\cos\alpha_i + \cos\beta_j}{2}\right)(1-\cos\alpha_i)^2}{1 - \frac{\cos 2\alpha_i + 4\cos\beta_j}{5}}(A-B)^{2m}.
\end{aligned}$$

Proof. From Eq. (7.36),

$$r\boldsymbol{\nu}_{ij}^{(2)} = \frac{1+\cos\alpha_i}{2}\boldsymbol{\nu}_{ij}^{(1)}, \tag{7.38}$$

where $i = 0,\ 1,\ \cdots,\ \frac{N}{2}-1$ and $j = 0,\ 1,\ \cdots,\ N-1$. Thus,

$$P_1^{-1}rQ_2\boldsymbol{\nu}_{ij}^{(2)} = \frac{4}{5}(1 - \omega + \frac{\omega(\cos\alpha_i + \cos\beta_j)}{2})(1+\cos\alpha_i)\boldsymbol{\nu}_{ij}^{(1)} \tag{7.39}$$

Moreover, since $\alpha_{N-i-1} = -\alpha_i + \pi$,

$$\sin k\alpha_{N-i-1} \sin l\beta_j = \begin{cases} \sin k\alpha_i \sin l\beta_j, & \text{if } k \text{ is odd} \\ -\sin k\alpha_i \sin l\beta_j, & \text{otherwise.} \end{cases} \tag{7.40}$$

Thus,

$$\begin{aligned} p\boldsymbol{\nu}_{ij}^{(1)} &= \cos\alpha_i \frac{\boldsymbol{\nu}_{ij}^{(2)} + \boldsymbol{\nu}_{N-i-1,j}^{(2)}}{2} + \frac{\boldsymbol{\nu}_{ij}^{(2)} - \boldsymbol{\nu}_{N-i-1,j}^{(2)}}{2} \\ &= \frac{1+\cos\alpha_i}{2}\boldsymbol{\nu}_{ij}^{(2)} - \frac{1-\cos\alpha_i}{2}\boldsymbol{\nu}_{N-i-1,j}^{(2)} \end{aligned} \tag{7.41}$$

Therefore,

$$\begin{aligned} B\boldsymbol{\nu}_{ij} &= \sum_{k=0}^{2m-1} \left\{1-\omega+\frac{\omega(\cos\alpha_i+\cos\beta_j)}{2}\right\}^k \omega\left(1-\frac{\cos\alpha_i+\cos\beta_j}{2}\right)\boldsymbol{\nu}_{ij} \\ &\quad + \omega\left(1-\frac{\cos\alpha_i+\cos\beta_j}{2}\right)\left(1-\omega+\frac{\omega(\cos\alpha_i+\cos\beta_j)}{2}\right)^m \frac{4}{5}(1+\cos\alpha_i) \\ &\quad \times \frac{1}{\omega\left(1-\frac{\cos 2\alpha_i+4\cos\beta_j}{5}\right)} \left\{\frac{1+\cos\alpha_i}{2}\left(1-\omega+\frac{\omega(\cos\alpha_i+\cos\beta_j)}{2}\right)^m \boldsymbol{\nu}_{ij}\right. \\ &\quad \left. -\frac{1-\cos\alpha_i}{2}\left(1-\omega+\frac{\omega(-\cos\alpha_i+\cos\beta_j)}{2}\right)^m \boldsymbol{\nu}_{N-i-1,j}\right\} \\ &= \left\{1-(A+B)^{2m}+\frac{2}{5}\frac{\left(1-\frac{\cos\alpha_i+\cos\beta_j}{2}\right)(1+\cos\alpha_i)^2}{1-\frac{\cos 2\alpha_i+4\cos\beta_j}{5}}(A+B)^{2m}\right\}\boldsymbol{\nu}_{ij} \\ &\quad - \frac{2}{5}\frac{\left(1-\frac{\cos\alpha_i+\cos\beta_j}{2}\right)(1-\cos^2\alpha_i)}{1-\frac{\cos 2\alpha_i+4\cos\beta_j}{5}}(A+B)^m(A-B)^m\boldsymbol{\nu}_{N-i-1,j}. \end{aligned}$$

Therefore, the theorem holds. □

The eigenvalues are calculated by the similar way as the one-dimensional case. Fig. 7.1 shows the eigenvalues solved in several ω, while optimal smoothing factor is $\frac{9}{25}$ for $\omega = \frac{4}{5}$. The minimum eigenvalue calculated here is closely to $1 - \frac{9}{25}(= \frac{16}{25})$ for $\omega = \frac{4}{5}$.

7.3.1.2 Standard coarsening

The stencils of r and p in the two-dimensional case are

$$[r]_k = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad \text{and} \quad [p^T]_k = \begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}, \tag{7.42}$$

where $k \in \Omega_1$, respectively. Thus,

$$r\boldsymbol{\nu}_{ij}^{(2)} = \frac{1+\cos\alpha_i+\cos\beta_j+\cos\alpha_i\cos\beta_j}{4}\boldsymbol{\nu}_{ij}^{(1)}, \tag{7.43}$$

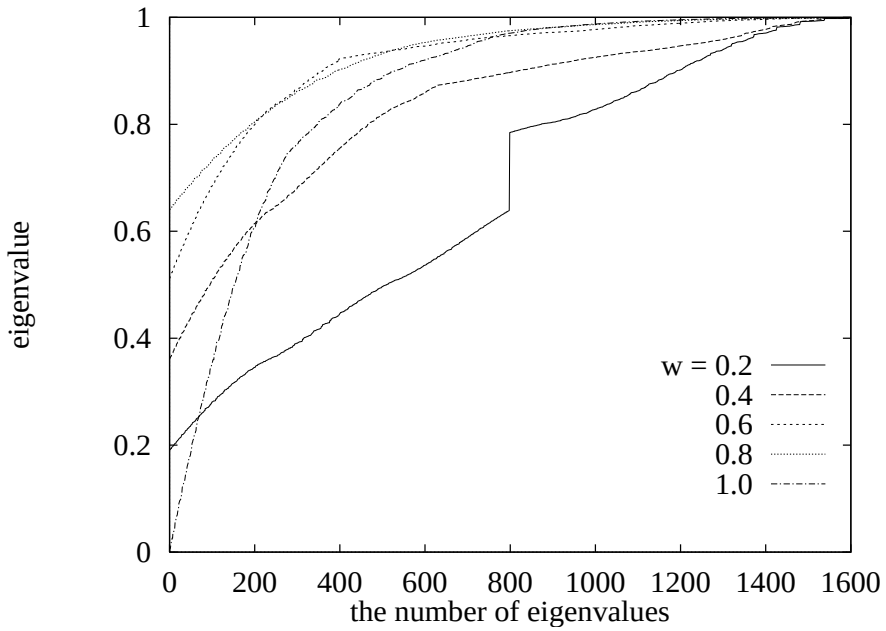

Figure 7.1: The distribution of eigenvalues of the two-grid preconditioned matrix with the damped Jacobi smoother and semicoarsening. Several curves mean different ω.

where $i,\ j = 0,\ 1,\ \cdots,\ \frac{N}{2}-1$. Thus,

$$P_1^{-1} r Q_2 \boldsymbol{\nu}_{ij}^{(2)} = \left(1 - \omega + \frac{\omega(\cos\alpha_i + \cos\beta_j)}{2}\right)(1+\cos\alpha_i)(1+\cos\beta_j)\boldsymbol{\nu}_{ij}^{(1)} \tag{7.44}$$

Moreover, $p\boldsymbol{\nu}_{ij}^{(1)}$ is represented by $\boldsymbol{\nu}_{ij}^{(2)}$, $\boldsymbol{\nu}_{N-i-1,j}^{(2)}$, $\boldsymbol{\nu}_{i,N-j-1}^{(2)}$ and $\boldsymbol{\nu}_{N-i-1,N-j-1}^{(2)}$,

$$\begin{aligned}
p\boldsymbol{\nu}_{ij}^{(1)} &= \frac{1}{4}(\boldsymbol{\nu}_{ij}^{(2)} - \boldsymbol{\nu}_{N-i-1,j}^{(2)} - \boldsymbol{\nu}_{i,N-j-1}^{(2)} + \boldsymbol{\nu}_{N-i-1,N-j-1}^{(2)}) \\
&\quad + \frac{\cos\alpha_i}{4}(\boldsymbol{\nu}_{ij}^{(2)} + \boldsymbol{\nu}_{N-i-1,j}^{(2)} - \boldsymbol{\nu}_{i,N-j-1}^{(2)} - \boldsymbol{\nu}_{N-i-1,N-j-1}^{(2)}) \\
&\quad + \frac{\cos\beta_j}{4}(\boldsymbol{\nu}_{ij}^{(2)} - \boldsymbol{\nu}_{N-i-1,j}^{(2)} + \boldsymbol{\nu}_{i,N-j-1}^{(2)} - \boldsymbol{\nu}_{N-i-1,N-j-1}^{(2)}) \\
&\quad + \frac{\cos\alpha_i\cos\beta_j}{4}(\boldsymbol{\nu}_{ij}^{(2)} + \boldsymbol{\nu}_{N-i-1,j}^{(2)} + \boldsymbol{\nu}_{i,N-j-1}^{(2)} + \boldsymbol{\nu}_{N-i-1,N-j-1}^{(2)}) \\
&= \frac{1+\cos\alpha_i}{2}\frac{1+\cos\beta_j}{2}\boldsymbol{\nu}_{ij}^{(2)} - \frac{1-\cos\alpha_i}{2}\frac{1+\cos\beta_j}{2}\boldsymbol{\nu}_{N-i-1,j}^{(2)} \\
&\quad - \frac{1+\cos\alpha_i}{2}\frac{1-\cos\beta_j}{2}\boldsymbol{\nu}_{i,N-j-1}^{(2)} + \frac{1-\cos\alpha_i}{2}\frac{1-\cos\beta_j}{2}\boldsymbol{\nu}_{N-i-1,N-j-1}^{(2)}
\end{aligned}$$

Therefore,

$$\begin{aligned}
B\boldsymbol{\nu}_{ij} &= \left\{1 - \left(1 - \omega + \frac{\omega(\cos\alpha_i + \cos\beta_j)}{2}\right)^{2m}\right\}\boldsymbol{\nu}_{ij} + \\
&\quad \frac{\left(1 - \frac{\cos\alpha_i + \cos\beta_j}{2}\right)\left(1 - \omega + \frac{\omega(\cos\alpha_i+\cos\beta_j)}{2}\right)^m (1+\cos\alpha_i)(1+\cos\beta_j)}{1 - \frac{\cos 2\alpha_i + \cos 2\beta_j}{2}}
\end{aligned}$$

$$\times \left\{ \frac{1+\cos\alpha_i}{2}\frac{1+\cos\beta_j}{2}\left(1-\omega+\frac{\omega(\cos\alpha_i+\cos\beta_j)}{2}\right)^m \boldsymbol{\nu}_{ij} \right.$$
$$-\frac{1-\cos\alpha_i}{2}\frac{1+\cos\beta_j}{2}\left(1-\omega+\frac{\omega(-\cos\alpha_i+\cos\beta_j)}{2}\right)^m \boldsymbol{\nu}_{N-i-1,j}$$
$$-\frac{1+\cos\alpha_i}{2}\frac{1-\cos\beta_j}{2}\left(1-\omega+\frac{\omega(\cos\alpha_i-\cos\beta_j)}{2}\right)^m \boldsymbol{\nu}_{i,N-j-1}$$
$$\left.+\frac{1-\cos\alpha_i}{2}\frac{1-\cos\beta_j}{2}\left(1-\omega+\frac{\omega(-\cos\alpha_i-\cos\beta_j)}{2}\right)^m \boldsymbol{\nu}_{N-i-1,N-j-1}\right\} \quad (7.45)$$

Let $k_1\boldsymbol{\nu}_{ij}+k_2\boldsymbol{\nu}_{N-i-1,j}+k_3\boldsymbol{\nu}_{i,N-j-1}+k_4\boldsymbol{\nu}_{N-i-1,N-j-1}$ be eigenvectors of the matrix B and λ be the corresponding eigenvalues. That is,

$$B(k_1\boldsymbol{\nu}_{ij}+k_2\boldsymbol{\nu}_{N-i-1,j}+k_3\boldsymbol{\nu}_{i,N-j-1}+k_4\boldsymbol{\nu}_{N-i-1,N-j-1}) =$$
$$\lambda(k_1\boldsymbol{\nu}_{ij}+k_2\boldsymbol{\nu}_{N-i-1,j}+k_3\boldsymbol{\nu}_{i,N-j-1}+k_4\boldsymbol{\nu}_{N-i-1,N-j-1}). \quad (7.46)$$

From Eq. (7.45) and (7.46),

$$\begin{pmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{pmatrix} \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{pmatrix} = \lambda \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{pmatrix}, \quad (7.47)$$

where $c_{ij}, (i,j=1,2,3,4)$ is the corresponding coefficient. Thus λ is equivalent to the eigenvalues of the 4 by 4 matrix of Eq. (7.47). This means we should solve the eigenvalue problem of 4 by 4 matrix or the characteristic equation of degree 4. Fig. 7.2 shows the distribution of eigenvalues solved numerically for several ω.

7.3.2 Red-black Gauss-Seidel smoother

For RB-GS smoother, we have the following theorem:

Theorem 7.3.2 *The eigenvalues of the two-grid preconditioned matrix with the RB-GS smoother are unity and roots of the following equation:*

$$\lambda^2+\left(-2+A^2+B^2-\frac{A^2(1-A^2)(1+A^2-B^2)^2}{4(1-A^2-B^2)}-\frac{B^2(1-B^2)(1-A^2+B^2)^2}{4(1-A^2-B^2)}\right)\lambda$$
$$+(1-A^2)(1-B^2)\left\{1+\frac{A^2(1+A^2-B^2)^2}{4(1-A^2-B^2)}+\frac{B^2(1-A^2+B^2)^2}{4(1-A^2-B^2)}\right\} = 0, \quad (7.48)$$

where $A=\frac{\cos\alpha_i+\cos\beta_i}{2}, B=\frac{-\cos\alpha_i+\cos\beta_i}{2}$ *and* $i,j=0,\cdots,\frac{N}{2}-1$.

Proof. Let $P^{-1}Q$ be the iteration matrix of the RB-GS smoother. We have

$$P^{-1}Q\boldsymbol{\nu}_{ij}=\frac{\cos\alpha_i+\cos\beta_j}{2}\frac{\boldsymbol{\nu}_{ij}+\boldsymbol{\nu}_{N-i-1,N-j-1}}{2}+\left(\frac{\cos\alpha_i+\cos\beta_j}{2}\right)^2\frac{\boldsymbol{\nu}_{ij}-\boldsymbol{\nu}_{N-i-1,N-j-1}}{2} \quad (7.49)$$

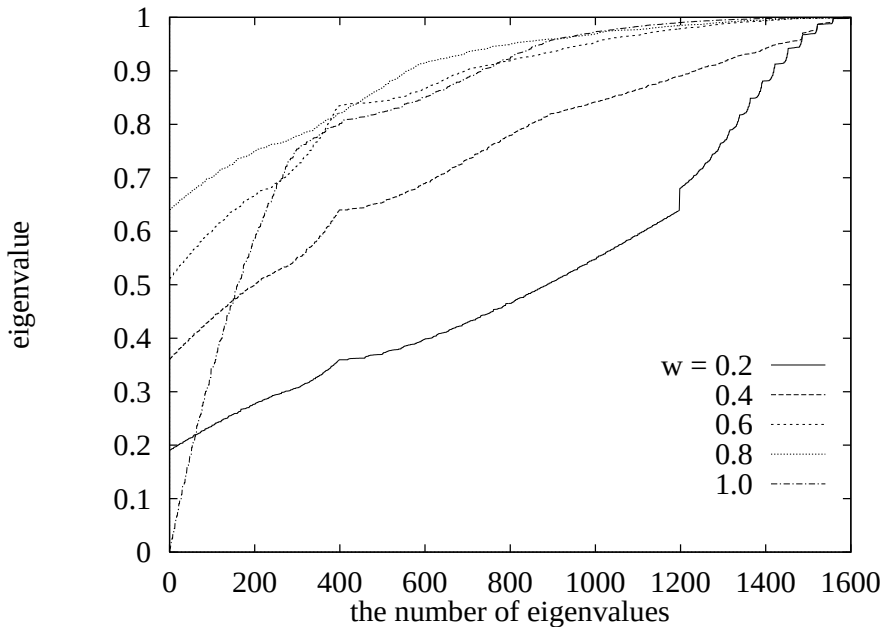

Figure 7.2: The distribution of eigenvalues of the two-grid preconditioned matrix with the damped Jacobi smoother and standard coarsening

Similarly,

$$\begin{aligned} P_1^{-1} r Q_2 \boldsymbol{\nu}_{ij} &= \frac{A(1+A^2-B^2)}{2}\left(2+\frac{\cos 2\alpha_i+\cos 2\beta_j}{2}\right)\boldsymbol{\nu}_{ij}^{(1)} \\ &\quad -\frac{A(1+A^2-B^2)}{2}\frac{\cos 2\alpha_i+\cos 2\beta_j}{2}\boldsymbol{\nu}^{(1)}_{\frac{N}{2}-i-1,\frac{N}{2}-j-1}, \end{aligned} \tag{7.50}$$

where $A=\frac{\cos\alpha_i+\cos\beta_i}{2}, B=\frac{-\cos\alpha_i+\cos\beta_i}{2}$. Thus,

$$\begin{aligned} B\boldsymbol{\nu}_{ij} &= \left\{1-\frac{A^2(1+A)}{2}+\frac{A^2(1-A^2)(1+A)(1+A^2-B^2)^2}{8(1-A^2-B^2)}\right\}\boldsymbol{\nu}_{ij} \\ &\quad -\frac{AB(1-A^2)(1+B)(1+A^2-B^2)(1-A^2+B^2)}{8(1-A^2-B^2)}\boldsymbol{\nu}_{N-i-1,j} \\ &\quad +\frac{AB(1-A^2)(1-B)(1+A^2-B^2)(1-A^2+B^2)}{8(1-A^2-B^2)}\boldsymbol{\nu}_{i,N-j-1} \\ &\quad +\left\{\frac{A^2(1-A)}{2}-\frac{A^2(1-A^2)(1-A)(1+A^2-B^2)^2}{8(1-A^2-B^2)}\right\}\boldsymbol{\nu}_{N-i-1,N-j-1} \end{aligned} \tag{7.51}$$

Here, again, we should solve 4 by 4 eigenvalue problem. Let C be the coefficient matrix similar to Eq. (7.47). Fortunately, two eigenvalues of the matrix C are unity, since $\mathrm{rank}(C-I_4)=2$ where I_4 is the identity matrix. Thus eigenvalues of C are roots of the following equation:

$$(\lambda-1)^2\left\{\lambda^2+\left(-2+A^2+B^2-\frac{A^2(1-A^2)(1+A^2-B^2)^2}{4(1-A^2-B^2)}-\frac{B^2(1-B^2)(1-A^2+B^2)^2}{4(1-A^2-B^2)}\right)\lambda\right.$$

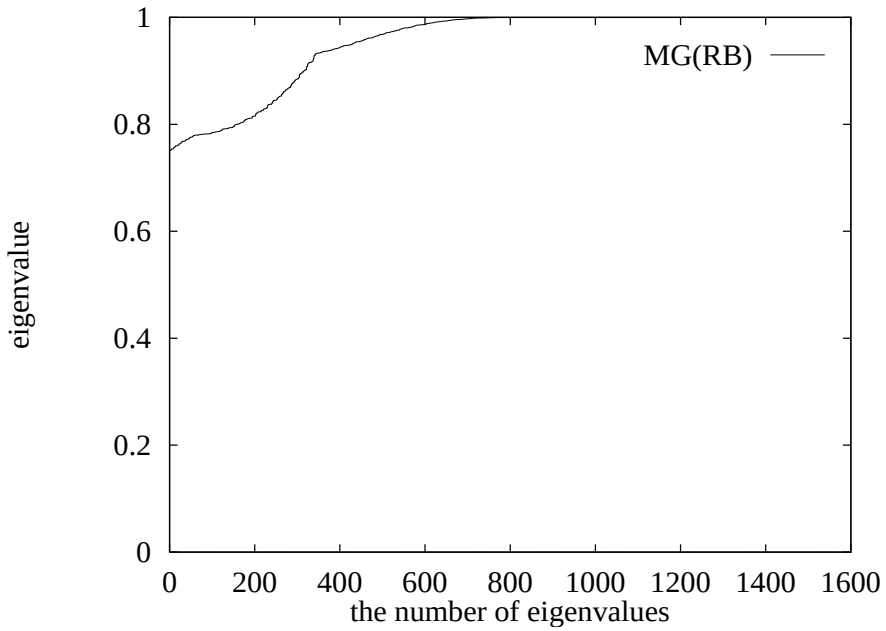

Figure 7.3: The distribution of eigenvalues of the two-grid preconditioned matrix with the Red-Black Gauss-Seidel smoother

$$+(1-A^2)(1-B^2)\left\{1+\frac{A^2(1+A^2-B^2)^2}{4(1-A^2-B^2)}+\frac{B^2(1-A^2+B^2)^2}{4(1-A^2-B^2)}\right\}\Bigg\} \quad = \quad 0,$$

where $i, j = 0, \cdots, \frac{N}{2} - 1$. Therefore the theorem holds. □

Fig. 7.3 shows the eigenvalues of the two-grid preconditioned matrix with RB-GS smoother for $N = 40$.

Lemma 7.3.3 *Let λ be eigenvalues of the two-grid preconditioned matrix with RB-GS smoother,*

$$\frac{3}{4} \leq \lambda \leq 1. \tag{7.52}$$

Proof. Since B is s.p.d., λ is real and positive. Let the left-hand term of Eq. (7.48) be $f(\lambda)$. From Fig. 7.4, the axis of the quadratic function $f(\lambda)$ is between 0.85 and 1. From Fig. 7.5, $f(1) \geq 0$ and $f(\frac{3}{4}) \geq 0$. Thus $\frac{3}{4} \leq \lambda \leq 1$. □

As described before, the Poisson equation with the Dirichlet boundary condition is similarly considered.

Theorem 7.3.4 *For 2-dimensional Poisson equation with the Dirichlet boundary condition, eigenvalues of the two-grid preconditioned matrix with RB-GS smoother are unity, $1 - \frac{\cos^2 \alpha_i}{4}$, $1 - \frac{\cos^2 \beta_i}{4}$ and roots of Eq. (7.48), where $\alpha_i = (i+1)\frac{\pi}{N}$ and $\beta_i = (i+1)\frac{\pi}{N}$ $(i,\ j = 0,\ 1,\ \ldots,\ \frac{N}{2} - 2)$.*

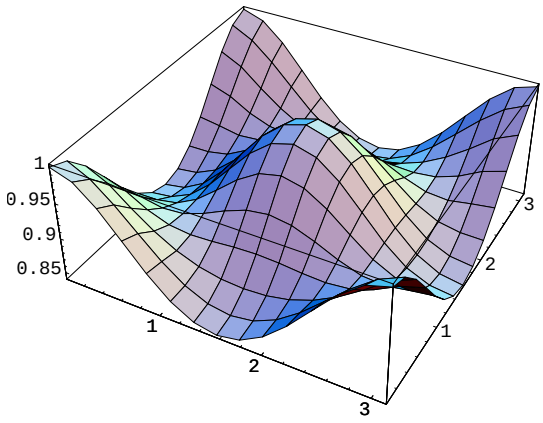

Figure 7.4: Axis of $f(\lambda)$

Proof. Let

$$\boldsymbol{\nu}_{ij} = (\sin\alpha_i \sin\beta_j \;\; \sin 2\alpha_i \sin\beta_j \;\; \cdots \;\; \sin(N-1)\alpha_i \sin(N-1)\beta_j)^T, \tag{7.53}$$

it is shown by the similar way of Theorem 7.3.2 that the eigenvalues are roots of Eq. (7.48) for $i,\ j = 0,\ 1,\ \ldots,\ \frac{N}{2}-2$ replacing $N-i-1$ to $N-i-2$ and so on. When $i = \frac{N}{2}-1$, $\alpha_i = \alpha_{N-i-2} = \frac{\pi}{2}$, so that $\cos\alpha_i = \cos\alpha_{N-i-2} = 0$. Thus

$$B\boldsymbol{\nu}_{ij} = \left(1 - \frac{A^2(1+A)}{2}\right)\boldsymbol{\nu}_{ij} + \frac{A^2(1-A)}{2}\boldsymbol{\nu}_{i,N-j-2}, \tag{7.54}$$

where $A = \frac{\cos\beta_j}{2}$ $(j = 0,\ 1,\ \ldots,\ \frac{N}{2}-2)$. Therefore, eigenvalues of B whose corresponding eigenvectors span a subspace $\mathrm{Span}\{\boldsymbol{\nu}_{ij},\ \boldsymbol{\nu}_{i,N-j-2}\}$, are unity and $1-A^2$. Similarly, when $j = \frac{N}{2}-1$, the eigenvalues are unity and $1-A^2$, where $A = \frac{\cos\alpha_i}{2}$. Finally, when $i = j = \frac{N}{2}-1$, the eigenvalue is unity. □

For the Dirichlet boundary problem, Lemma 7.3.3 also holds, hence the condition number of the preconditioned matrix is unchanged and $\frac{4}{3}$.

7.4 Convergence Rate of the MGCG Method

As described in Section 4.2, the convergence factor of k CG iterations for $L_l\boldsymbol{x} = \boldsymbol{b}$ is estimated by

$$\frac{2c^k}{1+c^{2k}}, \tag{7.55}$$

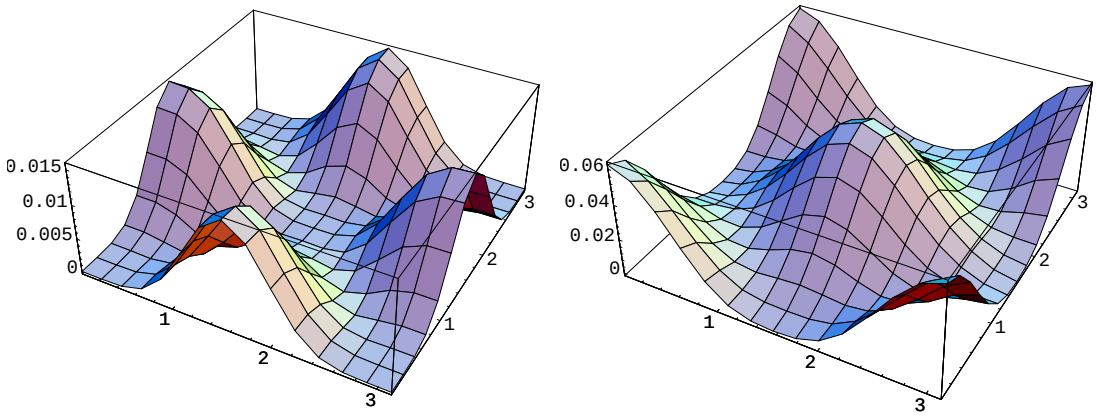

Figure 7.5: $f(1)$ & $f(3/4)$

Table 7.2: Asymptotic rate of convergence for 2-dimensional Poisson equation

	Jacobi	RB-GS
MG	0.444	0.602
MGCG	0.954	1.14

where $c = \frac{\sqrt{\sigma}-1}{\sqrt{\sigma}+1}$, $\sigma = \|L_l\| \, \|L_l^{-1}\|$, when L_l does not have isolated eigenvalues. Because the average convergence factor $c\left(\frac{2}{1+c^{2k}}\right)^{\frac{1}{k}}$ approaches to c as $k \to \infty$, asymptotic rate of convergence is estimated by $(-\log_{10} c)$.

As described before, the MGCG method and the MG method with the RB-GS smoother converge at one iteration for one-dimensional Poisson equation. For two-dimensional Poisson equation, the condition number of the two-grid preconditioned matrix with damped Jacobi smoother and RB-GS smoother are $\frac{25}{16}$ and $\frac{4}{3}$ respectively. The rate of convergence is independent of the number of unknowns, which is shown by Table. 7.2. In two dimensions, the rate of convergence of the MGCG method is 1.14, hence the convergence factor is less than 10^{-16} after 14 iterations. While this estimate is based on the condition number, it is shown that it is quite a practical estimate by numerical experiments in Appendix A. This estimate is also enough to show the MGCG method is superior to the MG method since the MGCG method needs only one more matrix-vector multiplication than the MG method in one iteration.

7.5 Multigrid Preconditioner

The multigrid preconditioned matrix is obtained by the two-grid preconditioned matrix replacing the coarse grid matrix with the multigrid preconditioning matrix on the coarse grid. The multigrid preconditioning matrix with l grid level is recursively defined by

$$\begin{aligned} M_1 &= L_1^{-1} \\ M_i &= (P^{-T}Q^T)^m \sum_{i=0}^{m-1} (P^{-1}Q)^i P^{-1} \\ &\quad + \sum_{i=0}^{m-1} (P^{-T}Q^T)^i P^{-T} + (P^{-T}Q^T)^m p M_{i-1} r (QP^{-1})^m \qquad (2 \le i \le l) \end{aligned} \tag{7.56}$$

The multigrid preconditioned matrix is defined by

$$B_l = M_l L_l. \tag{7.57}$$

Eigenvalues of B_l are expected to be analytically obtained by the similar way in the case of the two-grid preconditioned matrix, however it is quite complicated. Thus numerical solution for eigenvalues is used in the following analysis.

7.5.1 MG preconditioned matrix

In general, the multigrid preconditioned matrix B_l of Eq. (7.57) is nonsymmetric, hence using the Cholesky decomposition of the multigrid preconditioning matrix $M_l = U^T U$, consider an eigenvalue problem of the symmetric matrix UL_lU^T which is similar to M_lL_l and is the multigrid preconditioned matrix for the MGCG method. UL_lU^T is computed explicitly by an original matrix class library in C++.

In one dimension, MG preconditioner is proved as an exact solver for the Poisson equation, so it is not necessary to analyze numerically any more. In two dimensions, for the restriction of memory size, the computational domain is discretized into 32×32 meshes. Here we consider the eigenvalue problem of a 961×961 symmetric matrix, which occupies about 7 MB.

7.5.2 Eigenvalue analysis of MG preconditioned matrix by DCA

Fig. 7.6 shows the eigenvalue distribution of the MG preconditioned matrices whose coarse grid matrices are approximated by DCA. The top curve represents eigenvalues of the MG(2) preconditioned, that is, the two-grid preconditioned, matrix and the bottom one is that of the MG(5) preconditioned matrix which utilizes 5 grid levels. Since the computational domain is discretized into 32×32 meshes, the number of meshes on the coarsest grid of MG(5) are 2×2. The MGCG(2)

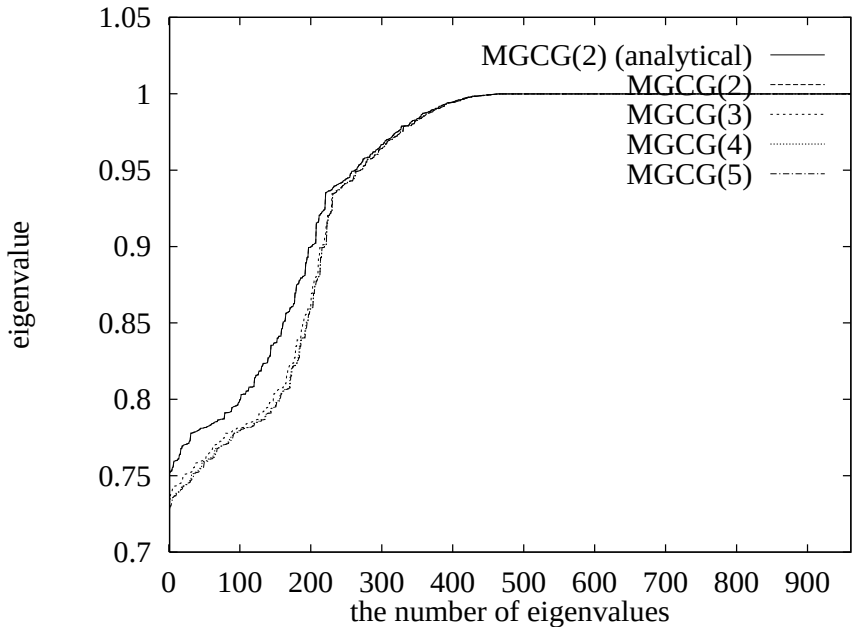

Figure 7.6: Eigenvalue distributions of multigrid preconditioned matrices by DCA with RB-GS smoother

curve seems to be one line but, in fact, two lines are overlapped: one is analytically calculated by Theorem 7.3.4 described in Subsection 7.3.2 and the other is obtained by the numerical computation of eigenvalues of a dense MG preconditioned matrix described in Subsection 7.5.1. Maximum error between these eigenvalues is less than 10^{-6}, which indicates the computation is quite reliable. The condition number of the MG(5) preconditioned matrix is slightly worse than that of the MG(2) preconditioned matrix, however, it is expected that the rate of convergence of MGCG(5) method is quite good and independent of the mesh size.

7.5.3 Eigenvalue analysis of MG preconditioned matrix by GCA

Eigenvalues and eigenvectors of the coarse grid matrix by DCA can be analyzed similarly as the fine grid matrix, while that of the coarse grid matrix by GCA cannot be. However, GCA automatically generates the coarse grid matrix from the fine grid matrix, so that it is indispensable for an algebraic MG method or a black box MG method that solves a system of linear equations using only given matrix not partial differential equation.

As described in Subsection 5.1.3, GCA generates the coarse grid matrix with nine-point pattern from the fine grid with five-point pattern in two dimensions. Thus, the smoother needs four-color

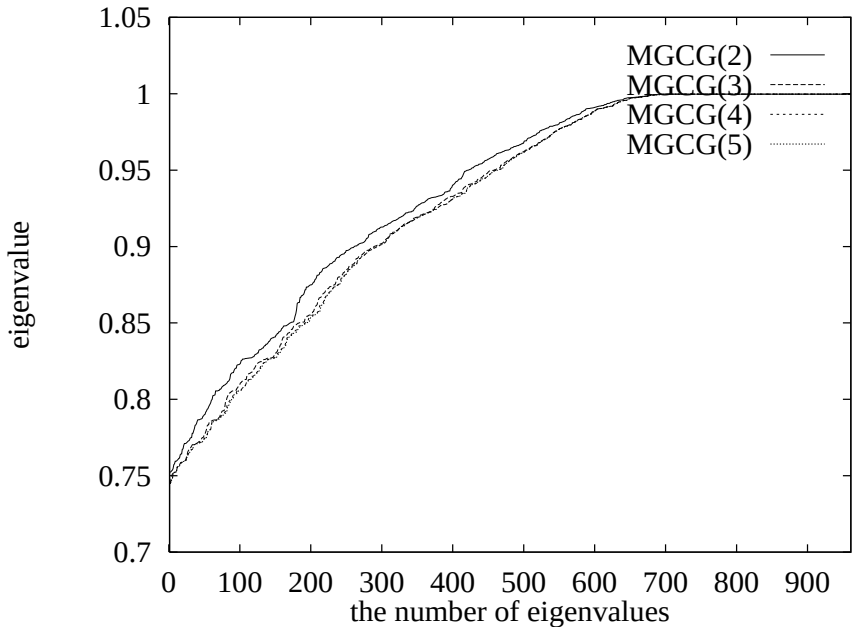

Figure 7.7: Eigenvalue distributions of multigrid preconditioned matrices by DCA with four-color GS smoother

ordering for high parallelism.

Fig. 7.7 and Fig. 7.8 show eigenvalues of the MG preconditioned matrices with four-color GS (fc-GS) smoother by DCA and GCA, respectively. In both figures, the top line represents eigenvalues of the MG(2) preconditioned matrix. By DCA, the condition number of MG(5) preconditioned matrix is slightly worse than that of MG(2), however, it is better than that of MG(5) preconditioned matrix with RB-GS. By GCA, the distribution of eigenvalues of MG(2) and MG(5) preconditioned matrix is changed, however, the condition numbers of these two matrices are nearly equal. Hence, upper bound of the rate of convergence of MGCG(5) method is nearly same as that of MGCG(2) method.

7.6 Related Works

Rate of Convergence of the two-grid method for two-dimensional model problem, which is Poisson equation in the unit square with the Dirichlet boundary condition, has been studied in early 1980s. Stüben and Trottenberg [51] have obtained the rate of convergence of the two-grid method with damped Jacobi smoother or RB-GS for the two-dimensional model problem. Using local Fourier modes, eigenvalues of the two-grid iteration matrix with the RB-GS smoother are obtained as roots

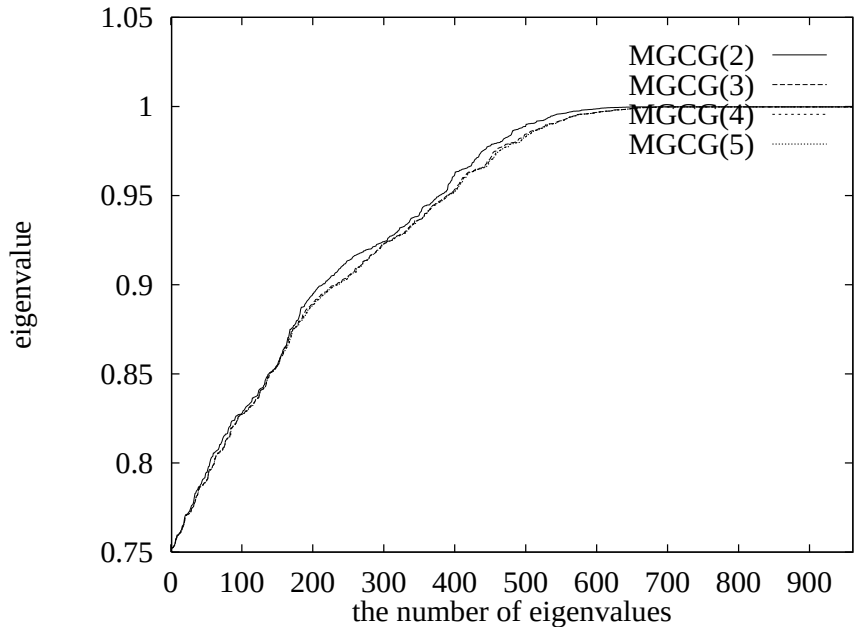

Figure 7.8: Eigenvalue distributions of multigrid preconditioned matrices by GCA with four-color GS smoother

of quadratic equations similarly to my thesis, because half of eigenvalues are zero. They uses RB-GS smoother for both pre- and post-smoothings, and showed the spectrum radius of the two-grid iteration matrix is 1/4 with one RB-GS iteration for either pre-smoothing or post-smoothing and $\frac{1}{2\nu}\left(\frac{\nu}{\nu+1}\right)^{\nu+1}$ with $\nu(=\nu_1+\nu_2)$ RB-GS iterations, however, they assumed $\boldsymbol{\nu}_i$ is an eigenvector of L, which is not satisfied in the model problem of my thesis with the Neumann-Dirichlet boundary condition. Ries et al. [45] analyzed the rate of convergence of MGR method that uses a rotated grid of mesh size $\sqrt{2}h$ between h and $2h$. They showed the spectral radius of the MGR-CH[ν] iteration matrix, which uses ν RB-GS pre-smoothing iterations and operator-dependent prolongation, is $\frac{1}{2}\frac{(2\nu)^{2\nu}}{(2\nu+1)^{2\nu+1}}$ for the two-dimensional model problem.

Fourier analysis is quite a useful tool for obtaining spectral radii of MG iteration matrices, however it is too complicated for MG method with three and more grids or three and more dimensional problems, thus the smoothing factor is analyzed in order to estimate the practical convergence factor. Brief explanation of the smoothing factor is described in Subsection 5.4.2. Wesseling [59] summarized smoothing factors of damped Jacobi, line Jacobi, point GS, line GS, ILU, RB-GS and zebra GS for rotated anisotropic diffusion equation and convection-diffusion equation. Yavneh [61] showed the smoothing factor of the RB-GS smoother for a class of elliptic operators has no

dependency on the dimension.

Chapter 8

MGCG METHOD FOR POISSON EQUATION WITH SEVERE COEFFICIENT JUMPS

In the previous chapter, the rate of convergence of the MGCG method is investigated for Poisson equation with constant diffusion coefficient. For strongly discontinuous coefficient, no theoretical result except a special case has been obtained. First, the special case is considered, and then complex case is analyzed numerically by eigenvalue analysis.

8.1 Special Problem

This section investigates a special problem whose MG preconditioned matrix has the same spectrum as that of the Poisson equation with constant diffusion coefficient.

8.1.1 One-dimensional problem

Consider one-dimensional Poisson equation

$$-\frac{d}{dx}k(x)\frac{du(x)}{dx} = f(x) \quad \text{with} \quad u(0) = u(1) = 0, \tag{8.1}$$

where

$$k(x) = \begin{cases} k_1, & \text{if } x \in (0,\ \frac{1}{2}) \\ k_2, & \text{otherwise} \end{cases} \tag{8.2}$$

Discretizing the range $(0,1)$ into N sections, a tridiagonal matrix appears, for example, the following matrix appears for $N=6$.

$$\frac{1}{h^2}\begin{pmatrix} 2k_1 & -k_1 & & & & \\ -k_1 & 2k_1 & -k_1 & & & \\ & -k_1 & k_1+k_2 & -k_2 & & \\ & & -k_2 & 2k_2 & -k_2 & \\ & & & -k_2 & 2k_2 & \end{pmatrix}, \tag{8.3}$$

where $h=\frac{1}{N}$. Since this matrix is splitted by

$$\frac{1}{h^2}\begin{pmatrix} 2k_1 & -k_1 & \\ -k_1 & 2k_1 & -k_1 \\ & -k_1 & k_1 \end{pmatrix} + \frac{1}{h^2}\begin{pmatrix} & & \\ & k_2 & -k_2 \\ & -k_2 & 2k_2 \end{pmatrix}, \tag{8.4}$$

eigenvalues of the Jacobi iteration matrix $P^{-1}Q$ of the matrix are

$$\cos\alpha_i, \tag{8.5}$$

where $\alpha_i=(i+1)\frac{\pi}{N}$ $(i=0,\ 2,\ 4)$, considering additional Neumann condition $\frac{d}{dx}u(\frac{1}{2})=0$. Therefore three in five eigenvalues of the Jacobi iteration matrix are obtained. The rest of eigenvalues are obtained by the following analysis. Let

$$\boldsymbol{\nu}_i=(\sin\alpha_i\ \sin 2\alpha_i\ \cdots\ \sin(N-1)\alpha_i)^T, \tag{8.6}$$

where $\alpha_i=(i+1)\frac{\pi}{N}$, for $i=1,\ 3$,

$$P^{-1}Q\boldsymbol{\nu}_i=\cos\alpha_i\boldsymbol{\nu}_i+\frac{k_1-k_2}{k_1+k_2}\sin\alpha_i\boldsymbol{e}_3, \tag{8.7}$$

where $\boldsymbol{e}_3=(0\ 0\ 1\ 0\ 0)^T$. Because of orthogonality of $\boldsymbol{\nu}_i$,

$$\boldsymbol{e}_3=\sum_{i=0}^{4}\boldsymbol{e}_3^T\boldsymbol{\nu}_i\,\boldsymbol{\nu}_i=\boldsymbol{\nu}_0-\boldsymbol{\nu}_2+\boldsymbol{\nu}_4. \tag{8.8}$$

Since $\boldsymbol{\nu}_0$, $\boldsymbol{\nu}_2$ and $\boldsymbol{\nu}_4$ are eigenvectors of $P^{-1}Q$, the rest of eigenvalues are

$$\cos\alpha_i, \tag{8.9}$$

where $\alpha_i=(i+1)\frac{\pi}{N}$ $(i=1,\ 3)$.

This discussion is applicable for general even N, and eigenvalues of the Jacobi iteration matrix are

$$\cos\alpha_i, \tag{8.10}$$

where $\alpha_i = (i+1)\frac{\pi}{N}$, which are equivalent to eigenvalues of the Jacobi iteration matrix for the Poisson equation with constant diffusion coefficient. Hence, the Jacobi method has the same rate of convergence.

For this special problem, we can obtain the spectrum of the two-grid preconditioned matrix. Because $\boldsymbol{\nu}_i$ $(i = 0,\ 2,\ \ldots,\ N-2)$ are eigenvectors of the Jacobi iteration matrix, after the same discussion of the proof of Theorem 7.2.4, eigenvalues of the two-grid preconditioned matrix are unity and

$$1 - \frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m} - \frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^{2m}.$$

From Eq. (8.7) and (8.8), for $i = 1,\ 3,\ \ldots,\ N-3$,

$$\begin{aligned} B\boldsymbol{\nu}_i &= \left(1 - \frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m}\right)\boldsymbol{\nu}_i \\ &\quad - \frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^m(1-\omega-\omega\cos\alpha_i)^m\boldsymbol{\nu}_{N-i-2} \\ &\quad + \sum_{0\le 2j<N-1} c_{2j}\boldsymbol{\nu}_{2j}. \end{aligned} \tag{8.11}$$

Since there are $\frac{N}{2}$ eigenvectors of B in $\mathrm{Span}\{\boldsymbol{\nu}_0,\ \boldsymbol{\nu}_2,\ \ldots,\ \boldsymbol{\nu}_{N-2}\}$, let two eigenvectors of B in $\mathrm{Span}\{\boldsymbol{\nu}_i,\ \boldsymbol{\nu}_{N-i-2}\}$ be $k_1\boldsymbol{\nu}_i + k_2\boldsymbol{\nu}_{N-i-2}$, then the corresponding eigenvalues are unity and

$$1 - \frac{1-\cos\alpha_i}{2}(1-\omega+\omega\cos\alpha_i)^{2m} - \frac{1+\cos\alpha_i}{2}(1-\omega-\omega\cos\alpha_i)^{2m}.$$

Therefore, the two-grid preconditioned matrix for the special problem has same spectrum of the two-grid preconditioned matrix for Eq. (7.3).

For RB-GS smoother, it is shown that eigenvalues of the two-grid preconditioned matrix are unity by the similar discussion. Hence, the following theorem holds.

Theorem 8.1.1 *For the special problem described in this section, the MG and MGCG method with the RB-GS smoother converges in one iteration for any $\boldsymbol{x}_0$, and the MGCG method with damped Jacobi smoother converges in two iterations for any $\boldsymbol{x}_0$.*

Proof. Because the MG preconditioned matrix has the same spectrum of the MG preconditioned matrix for the Poisson equation with uniform diffusion constant, the proof is complete. □

Note that this analysis holds for any k_1 and k_2 if ignoring rounding-off error.

8.1.2 Two-dimensional problem

Consider two-dimensional Poisson equation

$$-\nabla\,(k\nabla u) = f \quad \text{in } \mathcal{D} = (0,\ 1)\times(0,\ 1) \tag{8.12}$$

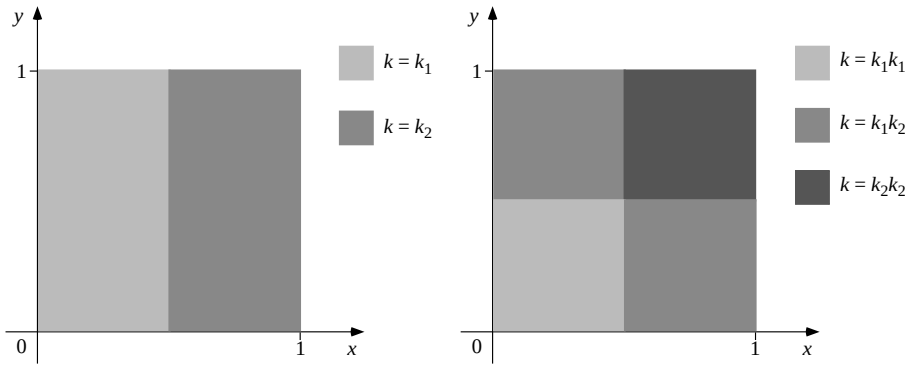

Figure 8.1: Diffusion constant of special problems

with k a diffusion coefficient. From the previous section, spectrum of the Jacobi iteration matrix is not changed even if diffusion coefficient of right half of the domain is different from that of left half. If let

$$\boldsymbol{\nu}_{ij} = (\sin\alpha_i \sin\beta_j \;\; \sin 2\alpha_i \sin\beta_j \;\; \cdots \;\; \sin(N-1)\alpha_i \sin(N-1)\beta_j)^T, \tag{8.13}$$

where $\alpha_i = (i+1)\frac{\pi}{N}$ and $\beta_i = (i+1)\frac{\pi}{N}$ $(i,\ j = 0,\ 1,\ \ldots,\ N-2)$, then it is shown the spectrum of the MG preconditioned matrix is equivalent to that in the Poisson problem with uniform diffusion coefficient even if diffusion coefficient of the two-dimensional Poisson equation is given by Fig. 8.1. Therefore the MGCG method has the same rate of convergence for the Poisson equation with uniform diffusion coefficient.

8.2 More Complex Problem

The special problem described in the previous section can be investigated analytically, however, for more complex diffusion constant, it is quite costly to solve eigenvalues of the iteration matrix by the similar way. Consider the following, not too complex, model problem

$$-\nabla\,(k\nabla u) = f \quad \text{in } \mathcal{D} = (0,\ 1) \times (0,\ 1) \tag{8.14}$$

with k a diffusion coefficient depicted by Figure 8.2. The boundary condition is given by

$$u = 0 \quad \text{on } \partial\mathcal{D}, \tag{8.15}$$

where $\partial\mathcal{D}$ denotes the boundary of the domain $\mathcal{D}$.

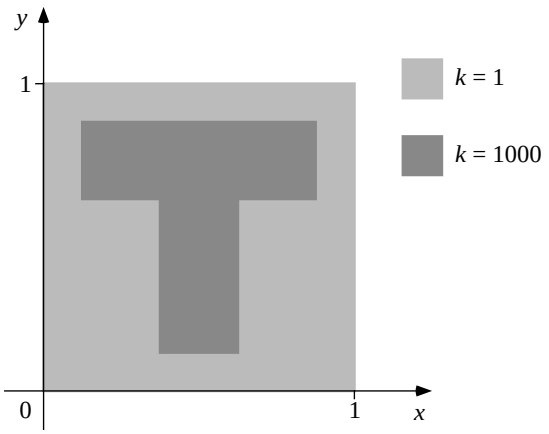

Figure 8.2: Diffusion constant of the model problem

8.2.1 Eigenvalue Distribution

The multigrid preconditioned matrix is computed by the same way described in Subsection 7.5.1. The eigenvalue distributions preconditioned by MG(2) and MG(5) are shown by Fig. 8.3. The MG(2) preconditioned matrix has nearly same distribution of eigenvalues as that of MG(2) preconditioned matrix for uniform diffusion constant problem. The MG(5) preconditioned matrix has three isolated eigenvalues except between $\frac{3}{4}$ and 1. The smallest eigenvalue is near zero, thus it degenerates the rate of convergence of the MG(5) method. However, MGCG(5) is expected to converge at three more iterations than that of MGCG(2). In fact, because the minimum eigenvalue is too close to zero, a few more iterations are necessary to converge [29, 5].

From these numerical results, the spectrum of the MG preconditioned matrix is nearly same as that for uniform diffusion constant problem, unless the problem cannot be discretized or approximated correctly on the coarsest grid. Otherwise, the MG preconditioned matrix has a few isolated eigenvalues like the MG(5) preconditioned matrix of the model problem, and the number of isolated eigenvalues is expected to depend on the mese size of the coarsest grid that can approximate correctly the problem. This proposition does not have a proof, however, it is supported by several numerical experiments and the special case on whose coarsest grid the problem is correctly approximated.

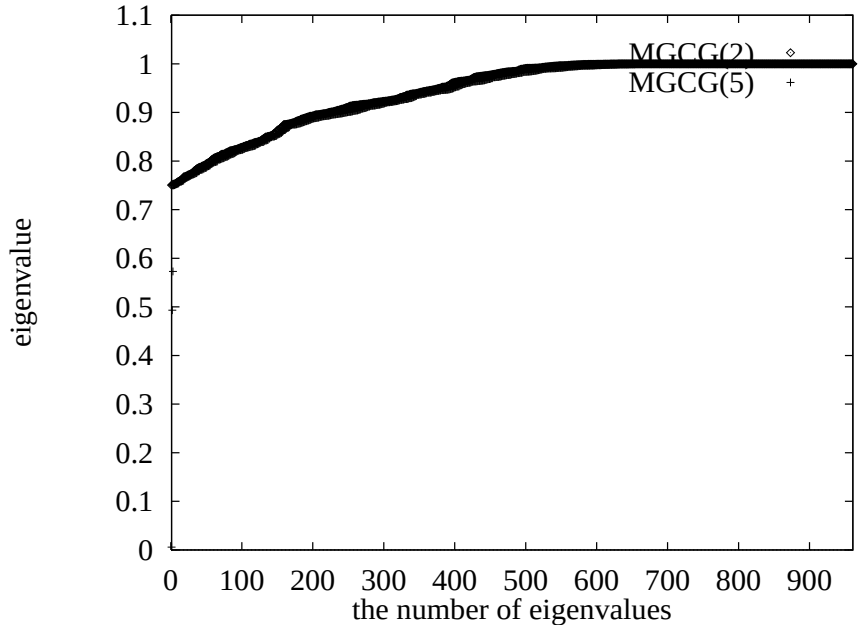

Figure 8.3: Eigenvalue distributions of two-grid preconditioned matrix and multigrid preconditioned matrix. The multigrid preconditioned matrix has three isolated eigenvalues.

Chapter 9

PARALLELIZATION OF THE MGCG METHOD

This chapter studies a parallelization of the MGCG method and its efficient implementation on distributed memory machines.

9.1 Parallelization of the MGCG Method

The MGCG method combines the CG method and the MG method, so it is necessary that both methods are efficiently parallelized and implemented for achieving high performance on parallel machines. The CG method consists of matrix-vector multiply, inner product and vector addition with multiplying by a scalar called `daxpy` in BLAS [36, 13]. The MG method consists of smoothing method, matrix-vector multiply, restriction and prolongation. Vector addition has $O(n)$ data-parallelism for n dimensional vector, and inner product has $O(n/\log n)$ data-parallelism, thus parallel steps are $O(n/p)$ and $O(n/p + \log p)$ respectively with p processors.

Restriction and prolongation have $O(n)$ data-parallelism similar to vector addition, and $O(n/p)$ parallel steps, however they need nearest-neighbor communication in computational domain, which interchanges boundary data between adjacent processors. Matrix-vector multiplication and smoothing method (damped Jacobi method, RB-GS, four-color GS and so on) have also $O(n)$ data-parallelism, and $O(n/p)$ parallel steps with nearest-neighbor communication.

9.1.1 Data distribution

Since required communication patterns are reduction and nearest-neighbor communication, a block (or tiling) data distribution is desirable to reduce communication. For simplicity, we only discuss the two-dimensional case, however, a similar argument is possible in three or more dimensions.

With p processors, consider the following (A) and (B) distributions:

(A) Vector (or computational domain) is distributed logically to $p \times 1$. This distribution is specified by the following directive in High Performance Fortran (HPF) [27].

```
!HPF$ DISTRIBUTE (BLOCK, *)
```

(B) Vector is distributed logically to $\sqrt{p} \times \sqrt{p}$. In HPF,

```
!HPF$ DISTRIBUTE (BLOCK, BLOCK)
```

In both distributions, the computational complexity is $O(n/p)$ with n the problem size, since it is proportional to the number of grid points in each processor. For nearest-neighbor communication, since the number of communication data is proportional to the number of grid points on the boundary between adjacent processors, that of the method (A) is $O(\sqrt{n})$ and that of the method (B) is $O(\sqrt{n}/\sqrt{p})$, while the number of communications of (A) is smaller than that of (B) since (A) only communicates both sides of the processor. Thus for fairly small p, (A) is advantageous, while (B) is preferable in the other case.

Ratio of communication to the computation of (A) and (B) are $O(\frac{p}{\sqrt{n}})$ and $O(\frac{\sqrt{p}}{\sqrt{n}})$ respectively, thus there is less communication overhead for n large.

9.1.2 Reducing the communication overhead

The communication overhead consists of network latency and message-handling overhead including the setup time and copy overhead between communication buffers. The message-handling overhead can not be avoided, however the network latency can be hidden by overlapping communication with computation. Figure 9.1 is an example of a fragment of block tri-diagonal matrix-vector multiplication. The computation can be separated into two parts: local computation that computes inner points in the subdomain and non-local computation that computes boundary points, since the communication pattern is described only by constant distance vector [2, 34]. The local computation does not need data owned by other processors, so can be performed locally. The non-local computation needs remote data, and is therefore not completed until receiving them. If remote data reach this processor during the local computation, the network latency can be hidden. The nodal code of Figure 9.1 in two processors is illustrated by Figure 9.2. For example, if `b[i][j]`, `a[i][j]` and `dd[i][j][0..4]` are allocated to the same processor, a block tri-diagonal matrix-vector multiplication needs the data `a[i - 1][j]`, `a[i][j - 1]`, `a[i][j + 1]` and `a[i + 1][j]`. In this case, the number of the inner points is $O(n)$ and the number of the boundary

```
for (i = 1; i < N - 1; i++)
    for (j = 1; j < M - 1; j++)
        b[i][j] = dd[i][j][0] * a[i - 1][j] +
                  dd[i][j][1] * a[i][j - 1] +
                  dd[i][j][2] * a[i][j] +
                  dd[i][j][3] * a[i][j + 1] +
                  dd[i][j][4] * a[i + 1][j];
```

Figure 9.1: Fragment of pentadiagonal matrix-vector multiplication

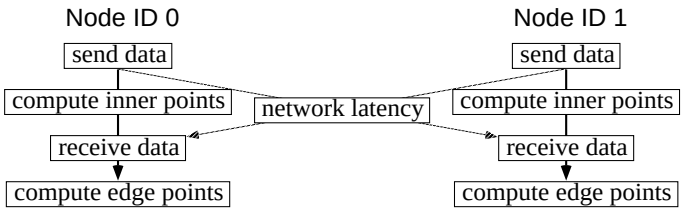

Figure 9.2: Nodal code for overlapping communication with computation

points is $O(\sqrt{n})$, where the allocated subdomain is discretized into n meshes. Thus the network latency is better hidden for n large. This technique is a popular and efficient optimizing technique for data-parallel compilers (for example [22, 34]).

9.1.3 Difficulties of parallelization

As described in Subsection 9.1.2, the network latency is hidden for n large. In this case, the ratio of communication to the total time is low, since the message-handling overhead is negligible to the total time. However, for n small, the network latency appears, i. e. the execution should wait to receive messages from other processors, and the ratio of communication becomes high, so the communication overhead is critical. Moreover, on quite coarse grid, the number of data-parallelism is smaller than the number of processors, so there are idle processors, while the computation on these coarse grids is necessary to gain the mesh-independent convergence property. Thus, in order that the parallel MGCG method achieves high performance, the following choices are promising.

1. The MG preconditioner exploits moderately coarse grid on which parallel efficiency is not lost. In this case, because it takes heavy cost to solve linear equations accurately on the coarsest grid, some approximate solution should be used.

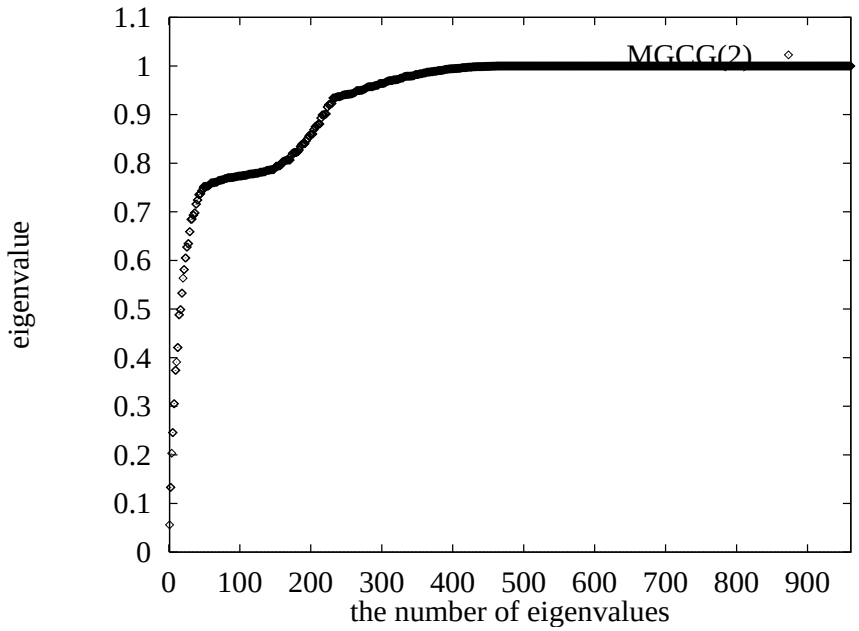

Figure 9.3: Eigenvalue distribution of two-grid preconditioned matrix with 1 iteration of RB-SGS on the coarsest grid

2. At the sacrifice of parallel efficiency, the MG preconditioner exploits very coarse grid.

In the first case, it is necessary to investigate the rate of convergence of the MGCG method with approximate solution on the coarsest grid because the MGCG method may need a lot of iterations until convergence. In the second case, though there is little parallelism on very coarse grid, the MGCG method converges quite fast independently of the mesh size as described in Chapters 7 and 8.

Fig. 9.3 shows the eigenvalue distribution of the MG(2) preconditioned matrix with one iteration of Red-Black symmetric GS (RB-SGS) as the approximate solution on the coarsest grid. Comparing with Fig. 7.6, there are 28 scattered eigenvalues, thus it is strongly desirable to solve more accurately for the two-grid preconditioner.

9.2 Evaluation of Trade-off

As described in the previous section, we have two promising choices that is a trade-off depending on the target parallel machines. This section evaluates the trade-off on Fujitsu highly parallel multicomputer AP1000 [28] and AP1000+ (AP+) [47]. The AP1000 is a distributed memory parallel machine with three kinds of networks: a two-dimensional torus network for communicating

among node processors which is called T-net, a ring network for mainly loading programs from a host to node processors called B-net and a tree network for fast synchronization called S-net. Each node has a 25MHz SPARC chip, a FPU whose peak speed is 5.56 MFlops in IEEE double precision, a 128 KB direct cache memory and a 64 MB local memory. Each port of the T-net has 25MB/s data transfer rate. AP+ is a successor of the AP1000 and its node processor is a 50MHz SuperSPARC chip. The AP+ uses the same networks as the AP1000, however it has a special hardware for useful active messages; PUT/GET active messages [57], which support direct remote-memory access.

To evaluate the trade-off, I have implemented a parallel MGCG method in C and message-passing library produced by Fujitsu. This program implements several optimizations: overlapping computation and communication as described in Subsection 9.1.2 and receiving messages out of order, for example, the matrix-vector multiply should receive messages from four processors in two-dimensional domain, however the order of arrival is non-determined, thus the message can be received in the arrival order not in the predetermined order. On quite coarse grid, the data-parallelism is less than the number of processors. I select a simple solution; all data is gathered into one processor, then the rest of computation is executed by the one processor, and then the computed data is spread to all processors. This strategy seems to be inefficient, however, it can be implemented simply and does not communicate any data on quite coarse grid, moreover, since there are small computation, computational time on quite coarse grid is expected to be negligible to the total computational time. The MGCG program is also tuned for fast execution; replacing floating-point division by multiplication of the inverse, eliminating unnecessary load/store and efficient cache utilization. Floating-point division is quite costly, so it should be avoided possibly.

To investigate the trade-off on AP1000 and AP+, consider the two-dimensional Poisson equation with Dirichlet boundary condition. Source term is randomly chosen. Stopping criteria is

$$\frac{\|\boldsymbol{r}_m\|_2}{\|\boldsymbol{r}_0\|_2} < 10^{-16}, \tag{9.1}$$

where $\boldsymbol{r}_m$ is the residual after m iterations. The following experiments are performed on 64 processors of AP1000 and 256 processors of AP+. The computational domain is discretized into $n \times n$ meshes, $n = 256$ for AP1000 and $n = 1024$ for AP+. Parallel MGCG method uses highly parallel RB-GS smoother.

9.2.1 Optimal grid level

First, we investigate optimal grid level of parallel V-cycle MGCG method with approximate solution on the coarsest grid. Table 9.1 shows the results on AP1000 and AP+ by the MGCG method

AP1000 (64 procs.; $n = 256$)

grids	#iter.	time (sec.)	MFlops
1	569	10.599	120.5
2	233	7.658	114.0
3	113	4.414	105.5
4	57	2.414	99.63
5	30	1.342	95.08
6	19	0.891	91.11
7*	16	0.778	88.03
8*	15	0.735	87.42

AP+ (256 procs.; $n = 1024$)

grids	#iter.	time (sec.)	MFlops
1	2260	64.628	1262.8
2	924	44.670	1247.5
3	445	23.856	1237.8
4	222	12.297	1225.5
5	112	6.316	1211.3
6	57	3.268	1194.8
7	30	1.747	1179.6
8*	19	1.128	1161.1
9*	16	0.953	1159.0
10*	15	0.894	1159.3

Table 9.1: The V-cycle MGCG method with various # of grid levels

with one RB-SGS iteration as the approximate solution on the coarsest grid, whose pre/post-smoother is one RB/BR-GS iteration. At the grid level with asterisk the computation is executed by one processor on very coarse grids.

As already shown by the eigenvalue analysis of Fig. 9.3, when the approximate solution on the coarsest grid is poor, many iterations are necessary for the MGCG method with small grid levels to converge. From this result, the MGCG method with many grid levels converges fast, while MFlops decreases by communication overhead and load imbalance. On AP1000, MFlops deteriorates 26% comparing 8 grid levels with one grid level, however the MGCG method with 8 grid levels whose coarsest grid is 2×2 converges fastest. On AP+, deterioration of MFlops is only 8%, so the MGCG method with 10 grid levels converges quite fast.

9.2.2 Optimal number of smoothing iterations

Next, we investigate the optimal number of iterations of the smoothing method. The results on AP1000 and AP+ are shown by Table 9.2. Greater number of smoothing iterations decreases the number of iterations until convergence, while MFlops decreases. That is because ratio of the MG preconditioner to the total computation becomes high.

Let ρ be the convergence factor of the MG with one smoothing iteration. When MG has two smoothing iterations, its convergence factor is expected to be ρ^2. Thus MG is expected to converge at half number of iterations. However, to use MG as a preconditioner of the CG method, the convergence factor of the MGCG method with one smoothing iteration and two smoothing

AP1000 (64 procs.; $n = 256$)

grids	6			7*			8*		
#sm.	#it.	time	MFlops	#it.	time	MFlops	#it.	time	MFlops
1	19	0.891	91.11	16	0.778	88.03	15	0.735	87.42
2	15	0.983	89.02	12	0.821	85.50	12	0.827	84.95
3	13	1.093	87.96	11	0.967	84.29	10	0.886	83.68
4	12	1.231	87.31	10	1.074	83.55	10	1.082	82.90

AP+ (256 procs.; $n = 1024$)

grids	8*			9*			10*		
#sm.	#it.	time	MFlops	#it.	time	MFlops	#it.	time	MFlops
1	19	1.128	1161.1	16	0.953	1159.0	15	0.894	1159.3
2	15	1.293	1092.6	12	1.038	1090.8	12	1.039	1089.9
3	14	1.581	1057.1	11	1.246	1055.3	10	1.135	1054.3
4	12	1.676	1034.9	10	1.402	1033.0	10	1.403	1031.7

Table 9.2: The V-cycle MGCG method with various # of smoothing iterations

iterations are given by

$$\frac{\sqrt{\frac{1}{1-\rho}}-1}{\sqrt{\frac{1}{1-\rho}}+1} \quad \text{and} \quad \frac{\sqrt{\frac{1}{1-\rho^2}}-1}{\sqrt{\frac{1}{1-\rho^2}}+1}, \tag{9.2}$$

respectively. However, the following inequality holds for $-1 < \rho < 1$:

$$\left(\frac{\sqrt{\frac{1}{1-\rho}}-1}{\sqrt{\frac{1}{1-\rho}}+1}\right)^2 \leq \frac{\sqrt{\frac{1}{1-\rho^2}}-1}{\sqrt{\frac{1}{1-\rho^2}}+1}, \tag{9.3}$$

equality holds if and only if $\rho = 0$. Therefore, the MGCG method does not converge at half number of iterations. In consequence, the MGCG method with one smoothing iteration converges faster than the MGCG method with two or more smoothing iterations.

9.2.3 Optimal MG schedule

This subsection investigates the W-cycle MGCG method that has much more coarse-grid corrections. From Table 9.3, for small grid levels, the W-cycle MGCG method converges faster than the V-cycle MGCG method with the same grid levels, however, for greater grid levels, the W-cycle MGCG method converges slower. That is because the W-cycle exploits coarser grids more frequently than finer grids, that is, the W-cycle exploits the coarsest grid most frequently in all the

AP1000 (64 procs.; $n = 256$)

grids	#iter.	time (sec.)	MFlops
1	569	10.599	120.5
2	181	6.495	110.5
3	64	3.318	93.04
4	24	1.660	76.03
5	15	1.420	57.84
6	15	2.094	39.89
7*	15	2.784	30.20
8*	15	2.879	29.26

AP+ (256 procs.; $n = 1024$)

grids	#iter.	time (sec.)	MFlops
1	2260	64.628	1262.8
2	716	37.438	1220.9
3	251	16.225	1201.8
4	90	6.612	1152.2
5	32	2.702	1045.6
6	16	1.644	879.2
7	15	2.054	665.9
8*	15	3.144	436.7
9*	15	3.372	407.8
10*	15	3.439	400.1

Table 9.3: The W-cycle MGCG method with various # of grid levels

AP1000 (64 procs.; $n = 256$)

grids	#it. (Ω_1)	#iter.	time	MFlops
5	13	16	0.833	82.55
6	8	15	0.764	84.11
7*	2	15	<u>0.731</u>	87.94
8*	1	15	0.735	87.42

AP+ (256 procs.; $n = 1024$)

grids	#it. (Ω_1)	#iter.	time	MFlops
7	13	16	0.977	1130.9
8*	9	15	0.896	1156.8
9*	3	15	0.8945	1158.9
10*	1	15	<u>0.8942</u>	1159.5

Table 9.4: The fastest MGCG method in each grid level

other (fine) grids, which has critical communication overhead and load imbalance.

9.2.4 More evaluation of trade-off

From the previous evaluation, it is shown that the V-cycle MGCG method with more accurate approximate solution, i. e. more iterations of a symmetric iterative method, on the coarsest grid is efficient. Table 9.4 shows calculation time and MFlops of the fastest MGCG method in each grid level on AP1000 and AP+. In the same grid level, the MGCG method converges fast by solving more accurately on the coarsest grid, however MFlops degenerates. That is because the computation on the coarsest grid has poor performance.

Consequently, when $n = 1024$, the MGCG method of ten grid levels is fastest on AP+, and when $n = 256$, that of seven grid levels with two RB-SGS iterations on the coarsest grid is fastest on AP1000. Because the MGCG method is an iterative method, the fastest MG schedule is slightly changed depending on stopping criteria, however, it is generally V-cycle of d grid levels for $n = 2^d$.

AP1000 ($n = 256$)

#procs.	1	4	16	64
#iter.	15			
time (s.)	51.5	14.6	3.27	0.73
MFlops	1.25	4.42	19.6	87.9
speedup	-	3.54	15.7	70.4
effic.	-	.88	.98	1.10

AP+ ($n = 256$)

#procs.	1	4	16	64	256
#iter.	15				
time (s.)	11.7	3.07	0.88	0.27	0.12
MFlops	5.48	20.9	73.4	235.8	547.7
speedup	-	3.83	13.4	43.1	100.0
effic.	-	.96	.84	.67	.39

Table 9.5: Performance of the parallel MGCG method in fixing n

9.3 Performance Evaluation

Performance of the parallel MGCG method is measured by speedup based on calculation time and *million floating-point operations per second* (MFlops). With p processors, time speedup S_p^{time} and MFlops speedup S_p^{MFlops} are calculated by

$$S_p^{\text{time}} = \frac{T_1}{T_p} \quad \text{and} \quad S_p^{\text{MFlops}} = \frac{M_p}{M_1},$$

respectively, where T_i is the calculation time and M_i is MFlops with i processors. Parallel efficiency is defined by

$$E_p = \frac{S_p}{p}$$

in calculation time or MFlops.

The result of MGCG(8) for $n = 256$ is presented by Table 9.5 on AP1000 and AP+. On the AP1000, superlinear speedup is observed in 64 processors. That is because size of cache memory is 64 times larger than that in one processor. However, on the AP+, only 40% parallel efficiency is achieved in 256 processors. Since AP+ has 4 times faster processor than that of AP1000, communication overhead is critical for n small. For n large, the same superlinear speedup is also expected to be observed on AP+, however, because each node of AP+ has a very small 36K cache memory, it is not unfortunately observed.

When fixing n, the cache memory is larger as the number of processors is larger, thus superlinear speedup can be observed if the communication overhead is surpassed. However, in this case, as the number of processors increases, several factors relating the performance is changed: the data size allocated in each processor is smaller, and the data size to be communicated is also smaller. Thus the performance analysis is complicated. All we can know is how fast a problem is solved!

To make the performance analysis easier, fix n^2/p, that is, memory size in each processor. In this case, since the problem size increases proportionally to the number of processors, the speedup

AP1000 ($n^2/p = 64^2$)

#procs.	1	4	16	64
n	64	128	256	512
#iter.	15	15	15	15
time (s.)	2.68	3.08	3.28	3.30
MFlops	1.45	5.17	19.6	78.4
speedup	-	3.56	13.5	54.1
effic.	-	.89	.85	.85

AP+ ($n^2/p = 128^2$)

#procs.	1	4	16	64	256
n	128	256	512	1024	2048
#iter.	15	15	15	15	15
time (s.)	2.83	3.07	3.21	3.22	3.23
MFlops	5.61	20.9	80.5	321.8	1284.9
speedup	-	3.74	14.4	57.4	229.2
effic.	-	.93	.90	.90	.90

Table 9.6: Performance of the parallel MGCG method in fixing n^2 in each processor

size	#iter.	time (sec.)	MFlops	SCG/MGCG
512^2	2271	8.118	1680.3	28.6 times
1024^2	4521	63.85	1704.6	71.4 times

Table 9.7: Parallel performance of the SCG method

and the parallel efficiency are measured in MFlops. The results on AP1000 and AP+ are shown by Table 9.6. Though the cache effect is not expected, the parallel efficiency is achieved 90% and 85% on AP+ and AP1000 respectively. Though AP+ has the same network as the AP1000, AP+ has quite small message handling overhead [47], higher parallel efficiency is achieved. For $p = 4$, the parallel efficiency is slightly better, because all processors has the boundary of the computational domain where the communication is not necessary.

9.4 Comparison with the SCG Method

The SCG method is a CG method with diagonal scaling preconditioner that needs no communication, thus this is 100% vectorizable and parallelizable and is expected to be complete scalable. It has been reported that total convergence time on vector machines is shorter than that of the (M)ICCG method, though the SCG method needs many more iterations [23]. That is because the (M)ICCG method requires the solution of triangular systems, which is quite expensive on vector machines. Table 9.7 is the result comparing with the MGCG method on AP+. The SCG method has high parallelism, and it gains high parallel efficiency, however, its rate of convergence is too cheap and depends on the mesh size. For $n = 512$, the MGCG method converges 29 times faster than the SCG method, and for $n = 1024$, 71 times faster. Although the SCG method is perfectly parallelizable, a poor convergence rate is critical.

9.5 Comments about Computational Complexity and Parallelism

In Chapters 7 and 8, the MGCG method is shown to converge at a constant number of iterations, thus the complexity of the MGCG method is $O(n)$ with n unknowns. Note that the MG method does not converge independently of the mesh size for Poisson equation with severe coefficient jumps. This complexity is superior to the nested dissection [16] and MICCG [18], whose complexities are $O(n^{\frac{3}{2}})$ for two-dimensional problems. Moreover, in Section 9.3, the MGCG method achieves 90% parallel efficiency, thus the comparison of other methods: SCG, MICCG and so on, is not fair as described in Section 9.4. The MGCG method is quite a promising method with both good complexity and high parallelism.

9.6 Related Works

Parallelization of the multigrid method has been studied and several parallel multigrid methods have been implemented. One natural parallelization approach is governed by the grid partitioning principles. Sbosny [46] analyzed and implemented a parallel multigrid method using the domain decomposition ideas. This algorithm is mathematically equivalent to the sequential multigrid algorithm. Besides the domain decomposition, the standard multigrid method with a highly parallel smoother is easy to be parallelized, however, the convergence rate of this method degenerates on Poisson problems with severe coefficient jumps. For efficient convergence on such equations, it has been reported that line (or plane) relaxation and semicoarsening are effective [1, 11]. Unfortunately, since these techniques have poor and awkward parallelism, they are inefficient on distributed memory machines. Hempel and Lemke [25] reported the most robust version of parallel black box multigrid, which has alternating zebra plane relaxation, gains about 60% on Intel iPSC/2 with 32 nodes. Dendy [12] implemented a semicoarsening multigrid algorithm suitable for SIMD machines on the CM-2. His method requires half the amount of relaxation computation as that required in the full coarsening multigrid case, while it requires line or plane relaxation. An alternative way of achieving robustness is to use multiple coarse grids. Mulder [38] and Naik [39] proposed a multiple semicoarsed grid (MSG) algorithm and Overman [44] implemented this algorithm on distributed memory machines. This method has ample parallelism and relative robustness. However, the MSG algorithm needs a larger amount of computation than the ordinary multigrid method.

Ashby et al. [3] implemented parallel MGCG method with damped Jacobi smoother and semicoarsening for groundwater flow problem on Cray T3D, whose parallel efficiency achieved about 80%. Washio and Oosterlee [58] proposed two MG preconditioners; SMG-S (semicoarsened multigrid as smoother) and MG-S (multigrid as smoother) [40], to gain robustness for rotated anisotropic

diffusion equation, curvilinear stretched grids and so on, and applied these preconditioners to Bi-CGSTAB [55]. From several numerical experiments on NEC Cenju-3 multicomputer, Bi-CGSTAB methods with these MG preconditioners converge faster than that with standard MG preconditioner, however their performances are lower because of using many coarse grids on which communication overhead is critical.

Chapter 10

CONCLUDING REMARKS

This thesis has investigated the MGCG method. In Chapter 6, sufficient conditions of the MG preconditioner are given such that the MG preconditioning (or preconditioned) matrix is symmetric and positive definite. It is shown that the V-cycle MG preconditioning matrix is s.p.d. if a post-smoother is adjoint of a pre-smoother and convergent on each grid level. For the W-cycle or much more cycles, it is shown that an additional condition; the V-cycle MG is convergent, is necessary.

To investigate the rate of convergence, Chapter 7 analyzes the distribution of eigenvalues of the two-grid preconditioned matrix with damped Jacobi smoother and RB-GS smoother for 1- and 2-dimensional Poisson equations. For 1 dimension, the MGCG method with damped Jacobi smoother converges at two iterations and that with RB-GS smoother converges at one iteration. For 2 dimension, asymptotic rate of convergence of the MGCG method is 1.14 independently of the mesh size, which is pessimistic estimate based on the condition number. While estimation by the eigenvalue distribution is necessary for a sharp estimate of the rate of convergence, it is enough to show the MGCG method is superior to the MG method since the MGCG method needs only one more matrix-vector multiplication than the MG method in one iteration. For MG preconditioner, though analytical result can be obtained by the similar way to the two-grid preconditioner, it is quite complicated, thus the eigenvalue distribution is obtained numerically. When coarse grid matrices are approximated by DCA, the condition number of the MG preconditioned matrix is slightly worse as the grid level increases. In the case of GCA, the distribution is changed, however the condition number is not changed, thus upper bound of rate of convergence is same as that of the two-grid preconditioner.

In Chapter 8, the rate of convergence is investigated for the Poisson equation with severe coefficient jumps. For special cases, it is shown that the MG preconditioned matrix has the same spectrum as that of the Poisson equation with uniform diffusion coefficient, however, for a

more complex problem, though it is expected to be able to be obtained in a similar way, it is too complicated, thus another analysis tool is necessary to estimate the rate of convergence. From numerical experiments of a model problem, the spectrum of the MG preconditioned matrix is nearly same as that for uniform diffusion constant problem, unless the problem cannot be discretized or approximated correctly on the coarsest grid. Otherwise, the MG preconditioned matrix has a few isolated eigenvalues and the number of isolated eigenvalues is expected to depend on the mese size of the coarsest grid that can approximate correctly the problem. This proposition does not have a proof, however, it is supported by several numerical experiments and the special cases on whose coarsest grid the problem is correctly approximated. If this assumption is true, the MGCG method converges at a few more iterations dependently on problems than that for the model problem.

Chapter 9 investigates a parallelization of the MGCG method and its efficient implementation on distributed memory machines. On quite coarse grids, parallel efficiency is low due to the communication overhead and load imbalance, while the computation on these coarse grids is necessary to gain the mesh-independent convergence property. Evaluating the parallel MGCG method on AP1000 and AP+, it is shown the MGCG method using very coarse grids is efficient in spite of losing parallel efficiency on such coarse grids. Moreover, parallel efficiency of the MGCG method gains 90% on 256 processors of AP+.

Since the MGCG method has high parallelism and fast convergence, this method is a very promising method as the solution of a large-scale, sparse, symmetric and positive definite matrix on not only serial machines but on distributed memory machines. For widespread use, it is desirable to build software libraries of scalable MGCG method.

Appendix A

CONVERGENCE FACTOR

In Chapter 7, the rate of convergence of the MGCG method is analytically obtained. Figure A.1 shows the numerical results of the Poisson equation with the uniform diffusion coefficient and a random source factor, which includes asymptotic convergence factor and upper bound of the average convergence factor based on Eq. (7.55). MGCG(2) has the two-grid preconditioner with one RB/BR-GS smoothing iteration. Three curves denoted by 64, 128 and 256 indicate average convergence factors for 64×64, 128×128 and 256×256 unknowns, respectively. From this figure, it is also shown numerically that the MGCG method converges independently of the mesh size, and that its average convergence factor is almost equal to or below the asymptotic convergence factor.

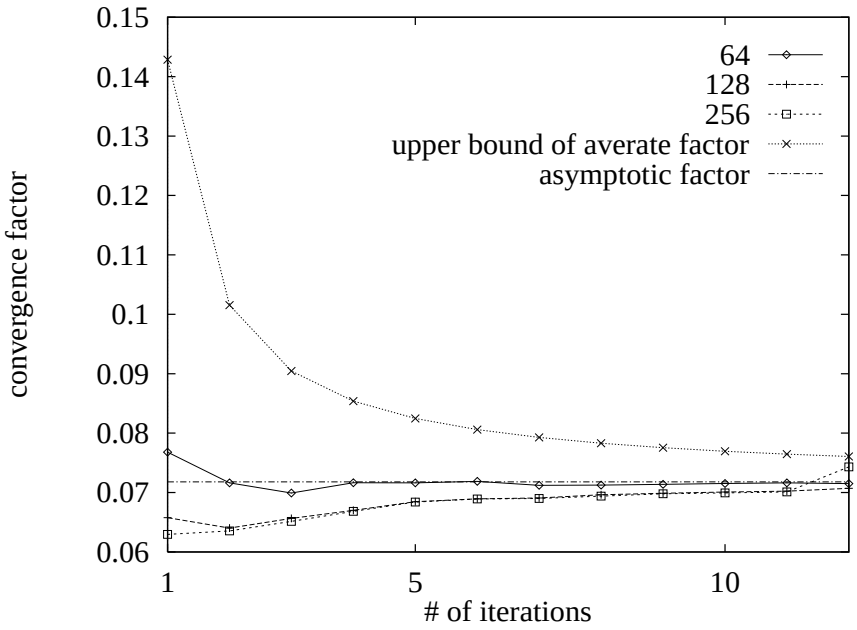

Figure A.1: Average convergence factor of the MGCG(2) method

References

[1] Alcouffe, R. E., A. Brandt, J. E. Dendy Jr., and J. W. Painter, "The multi-grid method for the diffusion equation with strongly discontinuous coefficient," *SIAM J. Sci. Stat. Comput.*, vol. 2, no. 4, pp. 430–454, 1981.

[2] Amarasinghe, S. P. and M. S. Lam, "Communication Optimization and Code Generation for Distributed Memory Machines," in *Proceedings of ACM Conference on Programming Language Design and Implementation*, pp. 126–138, June 1993.

[3] Ashby, S. F., R. D. Falgout, S. G. Smith, and T. W. Fogwell, "Multigrid preconditioned conjugate gradients for the numerical simulation of groundwater flow on the CRAY T3D," in *Proceedings of ANS International Conference on Mathematics and Computations, Reactor Physics, and Environment Analyses*, 1995.

[4] Axelsson, O. and G. Lindskog, "On the eigenvalue distribution of a class of preconditioning methods," *Numer. Math.*, vol. 48, pp. 479–498, 1986.

[5] Axelsson, O. and G. Lindskog, "On the rate of convergence of the preconditioned conjugate gradient method," *Numer. Math.*, vol. 48, pp. 499–523, 1986.

[6] Axelsson, O. and N. Munksgaard, "Analysis of incomplete factorizations with fixed storage allocation," in *Preconditioning Methods, Theory and Applications* (D. J. Evans, ed.), (London), pp. 219–241, Gordon and Breach, 1983.

[7] Axelsson, O., *Iterative Solution Methods.* Cambridge University Press, 1994.

[8] Bank, R. E. and C. C. Douglas, "Sharp estimates for multigrid rates of convergence with general smoothing and acceleration," *SIAM J. Numer. Anal.*, vol. 22, no. 4, pp. 617–633, August 1985.

[9] Brandt, A., "Multi-level adaptive solutions to boundary value problems," *Math. Comput.*, vol. 31, pp. 333–390, 1977.

[10] Brandt, A., "Multi-level adaptive techniques (MLAT) for partial differential equations: ideas and software," in *Proc. Symposium on Mathematical Software* (J. Rice, ed.), (New York), pp. 277–318, Academic, 1977.

[11] Dendy Jr., J. E., "Black box multigrid," *J. Comp. Phys.*, vol. 48, pp. 366–386, 1982.

[12] Dendy Jr., J. E., M. P. IDA, and J. M. Rutledge, "A Semicoarsening multigrid algorithm for SIMD machines," *SIAM J. Sci. Stat. Comput.*, vol. 13, no. 6, pp. 1460–1469, November 1992.

[13] Dongarra, J. and E. Grosse, "Distribution of mathematical software via electronic mail," *Communications of the ACM*, vol. 30, pp. 403–407, 1987.

[14] Faddeev, D. K. and V. N. Faddeeva, *Computational Methods of Linear Algebra.* Freeman, 1963.

[15] Fedorenko, R. P., "The speed of convergence of one iterative process," *USSR Comput. Math. and Math. Phys.*, vol. 4, no. 3, pp. 227–235, 1964.

[16] George, A., "Nested dissection of a regular finite element mesh," *SIAM J. Numer. Anal.*, vol. 10, pp. 345–363, 1973.

[17] Gustafsson, I., "Stability and rate of convergence of modified incomplete Cholesky factorization methods," Tech. Rep. 79.02R, Department of Computer Sciences, Chalmers University of Technology, Goteborg, Sweden, 1979.

[18] Gustafsson, I., "A class of first order factorization methods," *BIT*, vol. 18, pp. 142–156, 1978.

[19] Gutknecht, M. H., "Variants of Bi-CGSTAB for matrices with complex spectrum," *SIAM J. Sci. Comput.*, vol. 14, pp. 1020–1033, 1993.

[20] Hackbusch, W., "On the convergence of multi-grid iterations," *Beiträge Numer. Math.*, vol. 9, pp. 213–239, 1981.

[21] Hackbusch, W., *Multi-Grid Methods and Applications.* Springer-Verlag, 1985.

[22] Hatcher, P. and M. Quinn, *Data-parallel programming on MIMD computers.* MIT Press, 1991.

[23] Hayami, K. and N. Harada, "The scaled conjugate gradient method and vector processors," in *Proceedings of the First International Conference on Supercomputing Systems*, (St. Petersburg, Florida), pp. 213–221, IEEE Computer Society, 1985.

[24] Hemker, P. W., "On the order of prolongations and restrictions in multigrid procedures," *J. Comp. Appl. Math.*, vol. 32, pp. 423–429, 1990.

[25] Hempel, R. and M. Lemke, "Parallel black box multigrid," in *Proceedings of the fourth Copper Mountain Conference on Multigrid Method* (J. Mandel et al, ed.), pp. 255–272, April 1989.

[26] Hestenes, M. R. and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *J. Res. Nat. Bur. Standards Sect.*, vol. 49, pp. 409–436, 1952.

[27] High Performance Fortran Forum, "High Performance Fortran language specification, version 1.0," Tech. Rep. CRPC-TR92225, Center for Research on Parallel Computation, Rice University, Houston, TX, 1992 (revised May 1993).

[28] Ishihata, H., T. Horie, S. Inano, T. Shimizu, and S. Kato, "An Architecture of Highly Parallel Computer AP1000," in *IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing*, pp. 13–16, May 1991.

[29] Jennings, A., "Influence of the eigenvalue spectrum on the convergence rate of the conjugate gradient method," *J. Inst. Math. Appl.*, vol. 20, pp. 61–72, 1977.

[30] Johnson, O. G. and G. Paul, "Optimal parameterized inverse preconditioning for conjugate gradient calculations," Tech. Rep. RC8644, IBM Research Center, 1981.

[31] Kershaw, D. S., "The incomplete Cholesky-conjugate gradient method for the iterative solution of systems of linear equations," *J. Comp. Phys.*, vol. 26, pp. 43–65, 1978.

[32] Kettler, R. and J. A. Meijerink, "A multigrid method and a combined multigrid-conjugate gradient method for elliptic problems with strongly discontinuous coefficients in general domains," Tech. Rep. 604, Shell Oil Company, 1981.

[33] Kettler, R., "Analysis and Comparison of Relaxation Schemes in Robust Multigrid and Preconditioned Conjugate Gradient Methods," in *Multigrid Methods* (W. Hackbusch and U. Trottenberg, eds.), vol. 960 of *Lecture Notes in Mathematics*, pp. 502–534, Springer-Verlag, 1982.

[34] Koelbel, C. and P. Mehrotra, "Compiling Global Name-Space Parallel Loops for Distributed Execution," *IEEE trans. of Parallel and Distributed Systems*, vol. 2, no. 4, pp. 440–451, October 1991.

[35] Lanczos, C., "Solutions of systems of linear equations by minimized iterations," *J. Res. Nat. Bur. Standards Sect.*, vol. 49, pp. 33–53, 1952.

[36] Lawson, C., R. Hanson, D. Kincaid, and F. Krogh, "Basic Linear Algebra Subprograms for FORTRAN usage," *ACM Trans. Math. Soft.*, vol. 5, pp. 308–325, 1979.

[37] Meijerink, J. A. and H. A. van der Vorst, "An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix," *Math. Comput.*, vol. 31, pp. 148–152, 1977.

[38] Mulder, W. A., "A new multigrid approach to convection problems," *J. Comp. Phys.*, vol. 83, pp. 303–323, 1989.

[39] Naik, N. H. and J. V. Rosendale, "The improved robustness of multigrid elliptic solvers based on multiple semicoarsened grids," *SIAM J. Numer. Anal.*, vol. 30, no. 1, pp. 215–229, February 1993.

[40] Oosterlee, C. W., "The convergence of parallel multiblock multigrid methods," Tech. Rep. 850, Arbeitspapiere der GMD, 1994.

[41] Ortega, J., *Matrix Theory: A Second Course.* New York: Plenum Press, 1987.

[42] Ortega, J. M., *Introduction to Parallel and Vector Solution of Linear Systems.* Plenum Press, 1988.

[43] Ortega, J. M. and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables.* Orlando, Fla.: Academic, 1970.

[44] Overman, A. and J. V. Rosendale, "Mapping robust parallel multigrid algorithms to scalable memory architectures," in *Proceedings of Sixth Copper Mountain Conference on Multigrid Methods*, pp. 635–647, NASA Conference Publication 3224, April 1993.

[45] Ries, M., U. Trottenberg, and G. Winter, "A note on MGR methods," *Linear Algebra Appl.*, vol. 49, pp. 1–26, 1983.

[46] Sbosny, H., "Domain decomposition method and parallel multigrid algorithms," in *Multigrid Methods: Special Topics and Applications II* (W. Hackbusch and U. Trottenberg, eds.), no. 189 in GMD-Studien, pp. 297–308, May 1991.

[47] Shiraki, O., Y. Koyanagi, N. Imamura, K. Hayashi, T. Shimizu, T. Horie, and H. Ishihata, "Message Handling on the AP1000+ Highly Parallel Computer," in *Proceedings of JSPP'95*, pp. 233–240, May 1995. (In Japanese).

[48] Sonneveld, P., "CGS, a fast Lanczos-type solver for nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 10, pp. 36–52, 1989.

[49] Stein, P., "Some general theorem on iterants," *J. Res. Nat. Bur. Standards*, vol. 48, pp. 82–83, 1952.

[50] Stone, H. L., "Iterative solution of implicit approximations of multidimensional partial differential equations," *SIAM J. Numer. Anal.*, vol. 5, no. 3, pp. 530–558, 1968.

[51] Stüben, K. and U. Trottenberg, "Multigrid methods: Fundamental algorithms, model problem analysis and applications," in *Multigrid Methods, Proceedings of the Conference Held at Köln-Porz* (W. Hackbusch and U. Trottenberg, eds.), vol. 960 of *Lecture Notes in Mathematics*, pp. 1–176, Springer-Verlag, 1982.

[52] Tatebe, O., "The multigrid preconditioned conjugate gradient method," in *Proceedings of Sixth Copper Mountain Conference on Multigrid Methods*, pp. 621–634, NASA Conference Publication 3224, April 1993.

[53] Tatebe, O. and Y. Oyanagi, "Efficient implementation of the multigrid preconditioned conjugate gradient method on distributed memory machines," in *Proceedings of Supercomputing '94*, pp. 194–203, IEEE Computer Society, November 1994.

[54] van der Vorst, H. A., "A vectorizable variant of some ICCG methods," *SIAM J. Sci. Stat. Comput.*, vol. 3, no. 3, pp. 350–356, 1982.

[55] van der Vorst, H. A., "Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems," *SIAM J. Sci. Stat. Comput.*, vol. 13, pp. 631–644, 1992.

[56] Varga, R. S., *Matrix Iterative Analysis.* Prentice-Hall, 1962.

[57] von Eicken, T., D. E. Culler, S. C. Goldstein, and K. E. Schauser, "Active Messages: a Mechanism for Integrated Communication and Computation," in *Proceedings of the 19th International Symposium on Computer Architecture*, pp. 256–266, May 1992.

[58] Washio, T. and K. Oosterlee, "Experiences with robust parallel multilevel preconditioners for BiCGSTAB," Tech. Rep. 949, Arbeitspapiere der GMD, 1995.

[59] Wesseling, P., *An Introduction to Multigrid Methods.* John Wiley and Sons Ltd., 1992.

[60] Wittum, G., "Linear iterations as smoothers in multigrid methods: theory with applications to incomplete decompositions," *Impact Comput. Sci. Eng.*, vol. 1, pp. 180–215, 1989.

[61] Yavneh, I., "Multigrid smoothing factors for Red-Black Gauss-Seidel relaxation applied to a class of elliptic operators," *SIAM J. Numer. Anal.*, vol. 32, no. 4, pp. 1126–1138, August 1995.

[62] Young, D. P., *Iterative Solution of Large Linear Systems.* Academic Press, 1971.

[63] Zhang, S. L., "GPBi-CG: Generalized product-type methods based on Bi-CG for solving nonsymmetric linear systems," *SIAM J. Sci. Comput.*, vol. 18, no. 2, pp. 537–551, 1997.