AN INTRODUCTION TO MULTILEVEL METHODS

— Lecture Notes (draft), April 1998, ETHZ —

Professor Jinchao Xu
Department of Mathematics
The Pennsylvania State University
Univeristy Park, PA 16802
<xu@math.psu.edu>

and

ETHZ, HGG32.3
Rämistrasse 101
8092 Zürich
Phone: 41 1 6323347
<jxu@math.ethz.ch>

Contents

Chapter 1

Linear Vector Spaces

This chapter is devoted to some fundamentals of linear vector spaces.

1.1 Basic Consepts

Assume that $\mathcal{V}$ and W are two linear vector spaces over $\mathbb{R}$ or $\mathbb{C}$. Recall that a mapping[1] $A : \mathcal{V} \mapsto \mathcal{V}$ is said to be linear if

$$A(\alpha u + \beta \mathcal{V}) = \alpha A u + \beta A v \quad \forall\, u, v \in \mathcal{V}, \alpha, \beta \in \mathbb{R}.$$

The set of all linear operators from $\mathcal{V}$ to W will be denoted by $L(\mathcal{V}, W)$ and $L(\mathcal{V}) = L(\mathcal{V}, \mathcal{V})$.

1.2 Norms, inner products and spectrums

Definition 1.2.1 (Norm) *A norm on the vector space $\mathcal{V}$ is a function $\|\cdot\| : \mathcal{V} \mapsto \mathbb{R}$ that satisfies, for any $u, v \in \mathcal{V}$ and $\alpha \in \mathbb{R}$, the following three properties*

1. $\|v\| \geq 0$ *and* $\|v\| = 0$ *iff* $v = 0$*;*

2. $\|\alpha v\| = |\alpha| \|v\|$*;*

3. $\|u + v\| \leq \|u\| + \|v\|$.

(1.2.2) Example. $\mathcal{V} = \mathbb{R}^n$ and $p \geq 1$. Define

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}.$$

[1]Mapping, transformation and operator are synonyms in this book.

It can be verified that $\|\cdot\|$ satisfies all the three conditions in the above definition and hence it defines a norm in $\mathbb{R}^n$. Of the most important are $p = 1, 2$ and ∞:

$$\|x\|_1 = \sum_{i=1}^n |x_i|, \quad \|x\|_2 = \left(\sum_{i=1}^n |x_i|^2\right)^{\frac{1}{2}} \quad \text{and} \quad \|x\|_\infty = \max_{1\le i\le n} |x_i|.$$

Theorem 1.2.3 *Assume that $\|\cdot\|_1$ and $\|\cdot\|_2$ are given two norms on $\mathcal{V}$. Then there exist two positive constants c and C such that*

$$c\|v\|_1 \le \|v\|_2 \le C\|v\|_1.$$

Proof. Assume that $v_1, v_2, \cdots, v_n$ is a basis of $\mathcal{V}$. We define a function on $\mathbb{R}^n$ by $f(x) = \|v\|_2$ for $x \in \mathbb{R}^n$ such that $v = \sum_{i=1}^n x_i v_i$. It is straightforward to show that f is a positive, constinuous function on the compact set $S = \{x \in \mathbb{R}^n, \|v\|_1 = 1\}$. By the well-known property of continuous function, we have

$$0 < c = \min_{x\in S} f(x), \quad C = \max_{x\in S} f(x) < \infty.$$

Therefore $c \le \|v\|_2 \le C$ if $\|v\|_1 = 1$. Applying the above inequality with $v/\|v\|_1$ for general $v \ne 0$ then leads to the desired inequality. $\square$

Operator norms Notice that $\mathcal{L}(\mathcal{V})$ is a linear vector space, hence a norm of an operator $A \in L(\mathcal{V})$ can be defined by (1.2.1) in general.

Of the most useful is to define an operator norm by using a vector norm.

Proposition 1.2.4 *Assume that $\|\cdot\|$ is a norm on $\mathcal{V}$. Then*

$$\|A\| = \sup_{v\in\mathcal{V}} \frac{\|Av\|}{\|v\|} \quad \forall\, A \in L(\mathcal{V})$$

defines a norm on $L(\mathcal{V})$ that satisfies, for any $A, B \in L(\mathcal{V}), v \in \mathcal{V}$

$$\|Av\| \le \|A\|\|v\|$$

and

$$\|AB\| \le \|A\|\|B\|. \tag{1.2.5}$$

An operator norm $\|\cdot\|$ is said to be *submultiplicative* if it satisfies (1.2.5). A vector norm $\|\cdot\|_\alpha$ and an operator norm $\|\cdot\|_\beta$ is said to be consisitent if

$$\|Av\|_\alpha \le \|A\|_\beta \|v\|_\alpha \quad \forall\, A \in L(\mathcal{V}), v \in \mathcal{V}.$$

(1.2.6) Example. For the vector norms defined in (1.2), the corresponding matrix norms are given by

$$\|A\|_1 = \max_{1\le j\le n} \sum_{i=1}^n |a_{ij}|, \quad \|A\|_\infty = \max_{1\le i\le n} \sum_{j=1}^n |a_{ij}|.$$

The matrix norm $\|A\|_2$ will be given later.

We shall now give the definitions of inner products.

Definition 1.2.7 (Inner Product) *An inner product of $\mathcal{V}$ is a bilinear form $(\cdot,\cdot) : \mathcal{V} \times \mathcal{V} \mapsto \mathbb{R}$ such that for any $u, v, w \in \mathcal{V}$ and $\alpha, \beta \in \mathbb{R}$.*

1. *$(v, v) \geq 0$ and $(v, v) = 0$ iff $v = 0$;*
2. *$(u, v) = (v, u)$;*
3. *$(\alpha u + \beta v, w) = \alpha(u, w) + \beta(v, w)$*

(1.2.8) Example. For $\mathcal{V} = \mathbb{R}^n$. The Eucledean inner product is defined by

$$(x, y)_{l^2} = \sum_{i=1}^{n} x_i y_i.$$

Lemma 1.2.9 (Cauchy inequality) *Assume that $(\cdot,\cdot)$ is an inner product on $\mathcal{V}$, then*

$$|(u, v)| \leq \|u\|\|v\| \quad \forall\, u, v \in \mathcal{V}$$

where $\|v\| = (v, v)^{\frac{1}{2}}$ is a norm (induced by $(\cdot,\cdot)$) on $\mathcal{V}$.

Proof. By the definition of inner product, we have, for any $u, v \in \mathcal{V}$ and $t \in \mathbb{R}$

$$\|u\|^2 + t^2\|v\| - 2t(u, v) = (u - tv, u - tv) \geq 0.$$

Thus the desired inequality follows by taking $t = \|u\|/\|v\|$ (if $v \neq 0$). Moreover

$$\begin{aligned} \|u + v\|^2 &= (u + v, u + v) = \|u\|^2 + \|v\|^2 + 2(u, v) \\ &\leq \|u\|^2 + \|v\|^2 + 2\|u\|\|v\| \leq (\|u\| + \|v\|)^2. \end{aligned}$$

Thus $\|\cdot\|$ is indeed a norm. □

(1.2.10) Example. $\mathcal{V} = \mathbb{R}^n$. For the inner product given by (1.2), the Cauchy-Schwarz inequality gives

$$\left|\sum_{i=1}^{n} x_i y_i\right|^2 \leq \sum_{i=1}^{n} |x_i|^2 \sum_{i=1}^{n} |y_i|^2.$$

(1.2.11) Example. $\mathcal{V} = L^2(a, b)$, the vector space consisting of square interable functions over interval (a, b). Define an inner product by

$$(f, g)_{L^2} = \int_a^b f(x)g(x)dx.$$

The corresponding Cauchy-Schwarz inequality is

$$\left|\int_a^b f(x)g(x)dx\right|^2 \leq \int_a^b |f(x)|^2 dx \int_a^b |g(x)|^2 dx.$$

We shall review the concept of eigenvalue and eigenvectors. An eigenvalue of the operator $A \in L(\mathcal{V})$ is a real or complex number λ which, for some nonzero vector $v \in \mathcal{V}$, satisfies

$$(A - \lambda I)v = 0. \tag{1.2.12}$$

Any nonzero vector v satisfying the above equation is called an eigenvector of A corresponding to the eigenvalue λ.

If $A \in \mathbb{R}^{n\times n}$, obviously λ is an eigenvalue of A iff

$$\det(A - \lambda I) = 0.$$

We shall now give an example on the spectrum of symmetric tri-diagonal matrix of the following form:

$$A = \text{diag }(b, a, b)$$

namely $a_{ii} = a$ for $1 \le i \le N$ and $a_{i,i-1} = a_{i-1,i} = b$ for $2 \le i \le N$, and all other entries of A are zero.

Proposition 1.2.13 *The eigenvalues of the symmetric tri-diagonal matrix $A =$ diag (b, a, b), for any two real numbers a, b, are given by*

$$\lambda_k = a + 2b\cos\frac{k\pi}{N+1}, \quad 1 \le k \le N$$

and the eigenvectors $w_k, 1 \le k \le N$, are given by

$$w_{k,j} = \sin\left(\frac{jk\pi}{N+1}\right), \quad 1 \le j \le N$$

where $w_{k,j}$ is the jth component of the kth vector w_k. As a consequence, $\kappa(A) = O(h^{-2})$.

Proof. Assume $x \in \mathbb{R}^N$ is an eigenvector to an eigenvalue λ: $Ax = \lambda x$. Then

$$bx_{j-1} + ax_j + bx_{j+1} = \lambda x_j, \quad 1 \le j \le N \tag{1.2.14}$$

$$x_0 = x_{N+1} = 0. \tag{1.2.15}$$

The characteristic equation of the above finite difference equation is

$$b\mu^2 + (a - \lambda)\mu + b = 0.$$

It is easy to see any eigenvalue of A must satisfy $|\lambda - a| \le 2|b|$, the roots of the above equation are then given by

$$\mu = \frac{\lambda - a}{2b} \pm i\sqrt{1 - \left(\frac{\lambda - a}{2b}\right)^2} = e^{\pm i\theta}$$

where $i = \sqrt{-1}$ and θ is such that

$$\cos\theta = \frac{\lambda - a}{2b}, \quad \text{namely} \quad \lambda = a + 2b\cos\theta.$$

A solution of the difference equation (1.2.14) is given by

$$x_j = \mathrm{Im}[e^{i\theta}]^j = \sin j\theta.$$

In order that (1.2.15) to be satisfied, θ should be chosen so that

$$\sin(N+1)\theta = 0$$

which gives

$$\theta = \frac{k\pi}{N+1}, \quad 1 \le k \le N.$$

The desired result then follows easily. $\Box$

(1.2.16) Example. It is of interest to consider a special case that $a = 2$ and $b = -1$. In this case, we have

$$\lambda_k = 4\sin^2\frac{k\pi}{N+1}, \quad 1 \le k \le N.$$

As a consequence, we have $\kappa(A) \approx 4(N+1)^2$.

(1.2.17) EXERCISE. Any two symmetric tri-diagonal matrices are commutative.

Given $A \in L(\mathcal{V})$. We shall use the notation $\sigma(A)$ to denote the set of all eigenvalues of A. The spectral radius of A is defined and denoted by

$$\rho(A) = \max_{\lambda\in\sigma(A)} |\lambda|.$$

Proposition 1.2.18 *For any $A, B \in L(\mathcal{V})$, $\sigma(AB) = \sigma(BA)$ and $\sigma(A) = \sigma(T^{-1}AT)$ for any invertible $T \in L(\mathcal{V})$.*

Proposition 1.2.19 *Assume $\|\cdot\|$ is a submultiplicative norm on $L(\mathcal{V})$. Then*

$$\rho(A) \le \|A\|, \quad \forall\, A \in L(\mathcal{V}).$$

Proposition 1.2.20 *Assume that $A \in L(\mathcal{V})$, then*

$$\lim_{k\to\infty} A^k = 0 \quad \textit{iff} \quad \rho(A) < 1.$$

1.3 Symmetric Positive Definite Operators

We begin our discussion in the Euclidean space $\mathbb{R}^n$. An operator on $\mathbb{R}^n$ is a matrix, say $A = (a_{ij}) \in \mathbb{R}^{n\times n}$. When A is said to be self–adjoint or symmetric, it often means that $A = A^t$ (here "t" denotes the transposition), that is, $a_{ij} = a_{ji}$, or equivalently

$$(Ax, y)_{l^2} = (x, Ay)_{l^2} \quad \forall\, x, y \in \mathbb{R}^n,$$

where $(\cdot,\cdot)_{l^2}$ is given by (1.2). In general, we have

$$(Ax, y)_{l^2} = (x, A^t y)_{l^2} \quad \forall\, x, y \in \mathbb{R}^n.$$

Definition 1.3.1 *Given $A \in L(\mathcal{V})$, $B \in L(\mathcal{V})$ is called the adjoint of A with respect to the inner product $(\cdot,\cdot)$, if*

$$(Au, v) = (u, Bv).$$

We denote $B = A^t$. A is said to be self-adjoint or symmetric if $A^t = A$. A is said to be SPD (symmetric positive definite) if A is symmetric and $(Av, v) > 0$ for any $v \in \mathcal{V} \setminus \{0\}$.

Lemma 1.3.2 *Assume that $A \in L(\mathcal{V})$ is symmetric with respect to $(\cdot,\cdot)$. Then*

1. $\sigma(A) \subset \mathbb{R}$.
2. *If $Av_i = \lambda_i v_i$, $\lambda_1 \neq \lambda_2$, then $(v_1, v_2) = 0$.*
3. *A has a complete set of eigenvectors.*

(1.3.3) EXERCISE. Prove that A is self–adjoint if and only if A has all real eigenvalues and a complete set of eigenvectors (namely these eigenvector form a basis of $\mathcal{V}$).

Proposition 1.3.4 *Assume $A \in L(\mathcal{V})$ is SPD w.r.t $(\cdot,\cdot)$. Then*

1. *Assume that "t" and "$*$" denote the transpositions with respect to $(\cdot,\cdot)$ and $(\cdot,\cdot)_A$ respectively, then*
$$(BA)^* = B^t A.$$
2. *BA is symmetric with respect to $(\cdot,\cdot)_A$ iff B is symmetric with respect to $(\cdot,\cdot)$.*
3. *If B is SPD with respect to the same inner product $(\cdot,\cdot)$, BA is SPD self-adjoint with respect to either $(A\cdot,\cdot)$ or $(B^{-1}\cdot,\cdot)$.*

(1.3.5) EXERCISE. Assume that $A = (a_{ij})$ is SPD. Prove that $a_{ii} > 0$ and $|a_{ij} \le \sqrt{a_{ii}a_{jj}}$ for $1 \le i, j \le n$.

Proposition 1.3.6 *Assume $A \in L(\mathcal{V})$ is SPD w.r.t $(\cdot,\cdot)$. Then $\sigma(A) \subset (0, \infty)$, $(\cdot,\cdot)_A = (A\cdot,\cdot)$ defines another inner product on $\mathcal{V}$ and*

$$\lambda_{\min}(A) = \min_{v\in\mathcal{V}} \frac{(Av, v)}{\|v\|^2} \quad \textit{and} \quad \lambda_{\max}(A) = \max_{v\in\mathcal{V}} \frac{(Av, v)}{\|v\|^2}. \tag{1.3.7}$$

Proposition 1.3.8 *Assume that $A : \mathcal{V} \mapsto \mathcal{V}$ and $\|\cdot\|$ is induced by an inner product $(\cdot,\cdot)$ on $\mathcal{V}$. If A is symmetric with respect to $(\cdot,\cdot)$,*

$$\|A\| = \rho(A).$$

In general,

$$\|A\| = \rho(A^tA)^{\frac{1}{2}}.$$

Here A^t is the adjoint operator of A with respect to $(\cdot,\cdot)$.

(1.3.9) Example. Given $\gamma \in (0,1)$, consider the symmetric matrix

$$A = (\gamma^{|i-j|}) \in \mathbb{R}^{n\times n},$$

then

$$\rho(A) \le \|A\|_1 = \max_{1\le j\le n} \sum_{i=1}^{n} \gamma^{|i-j|} \le \frac{2}{1-\gamma}.$$

Thus

$$\sum_{i,j=1}^{n} \gamma^{|i-j|} x_i x_j \le \frac{2}{1-\gamma} \sum_{i=1}^{n} x_i^2.$$

In general

$$\sum_{i,j=1}^{\infty} \gamma^{|i-j|} x_i x_j \le \frac{2}{1-\gamma} \sum_{i=1}^{\infty} x_i^2. \tag{1.3.10}$$

(1.3.11) EXERCISE. If $A \in L(\mathcal{V})$ is self-adjoint with respect to $(\cdot,\cdot)$ and it is semi-definite, i.e. $(Av,v) \ge 0 \quad \forall\, v \in \mathcal{V}$. Then

$$(Au,v) \le (Au,u)^{\frac{1}{2}}(Av,v)^{\frac{1}{2}} \quad \forall\, u,v \in \mathcal{V}.$$

(1.3.12) EXERCISE. Under the assumption of Exercise 1.3, prove that, for any $\alpha \in [0,1]$

$$\|A^{\frac{1+\alpha}{2}}v\| \le (Av,v)^{\frac{1-\alpha}{2}}\|Av\|^{\alpha} \quad \forall\, v \in \mathcal{V}.$$

The following simple result is useful in multigrid analysis.

Proposition 1.3.13 *Assume that $K : \mathcal{V} \mapsto \mathcal{V}$ is a symmetric operator with $\sigma(K) \subset [0,1]$, then*

$$((I-K)K^m v, K^m v) \le \frac{1}{2m}(\|v\|^2 - \|K^m v\|^2).$$

Proof. The proof follows from the following elementary inequality

$$(1-x)x^{2m} \le 1 - x^{2m} \quad \forall\, x \in (0,1)$$

□

1.4 Schur complement

Given an SPD matrix in a block form:

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$$

A system of equation $Ax = b$ may be written as

$$\begin{aligned} A_{11}x_1 + A_{12}x_2 &= b_1 \\ A_{21}x_1 + A_{22}x_2 &= b_2 \end{aligned}$$

Soving for x_1 in terms of x_2 from the first equation and substituting the result to the second equation gives

$$Sx_2 = b_2 - A_{21}A_{11}^{-1}b_1$$

where

$$S = A_{22} - A_{21}A_{11}^{-1}A_{12}$$

is called the *Schur complement* of A_{11} of the matrix A.

Lemma 1.4.1 *The Schur complement S is still SPD.*

Lemma 1.4.2 *The condition number of an SPD matrix is greater or equal to the condition number of any of its Schur complement.*

Proof. Using the notation above, we need to show that

$$\kappa(A_{22} - A_{21}A_{11}^{-1}A_{12}) \le \kappa(A).$$

□

The concept of Schur complement is very useful in domain decomposition methods.

1.5 Matrix tensor products

In this section, we shall give a brief introduction of matrix tensor products which will be useful in the discussion of finite element equations on certain regular domains in two and three dimensions.

Definition 1.5.3 *Given $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times q}$, the tensor product of A and B, denoted by $A \otimes B$, is an $mp \times nq$ matrix or a $m \times n$ block matrix defined by*

$$A \otimes B = \begin{bmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \cdots & \cdots & \cdots & \cdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{bmatrix}$$

By definition we have

Theorem 1.5.4 *1.* $0 \otimes A = A \otimes 0 = 0$.

2. $(A_1 + A_2) \otimes B = A_1 \otimes B + A_2 \otimes B, A \otimes (B_1 + B_2) = A \otimes B_1 + A \otimes B_2$.

3. $(\alpha A) \otimes (\beta B) = (\alpha\beta) A \otimes B \quad \forall\, \alpha, \beta \in \mathbb{R}$.

4. $(A \otimes B) \otimes C = A \otimes (B \otimes C)$.

The following less obvious result is important.

Theorem 1.5.5 *1.* $(A_1 \otimes B_1)(A_2 \otimes B_2) = (A_1 A_2) \otimes (B_1 B_2)$.

2. $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$ *if both* A *and* B *are invertible.*

3. $(A \otimes B)^t = A^t \otimes B^t$.

4. The product of two lower (upper) triangluar matrices is still lower (upper) triangular.

5. The product of two orthogonal matrices is still orthogonal.

Proof. The first identity again follows from the definition and the second identity follows from the first one. The proof for other identities is also straightforward. □

Theorem 1.5.6 *Let* $\phi(s,t) = \sum_{i,j=0}^{p} \alpha_{ij} s^i t^j$, *define*

$$\phi(A, B) = \sum_{i,j=0}^{p} \alpha_{ij} A^i \otimes B^j.$$

If $\{\lambda_i : \ i = 1 : m\}$ *and* $\{\mu_j : \ j = 1 : n\}$ *are the spectral sets of* A *and* B *respectively, then the spectral set of* $\phi(A, B)$ *is* $\{\phi(\lambda_i, \mu_j) : \ i = 1 : m, j = 1 : n\}$. *Furthermore if* $Ax = \lambda x$ *and* $By = \mu y$, *then* $\phi(A, B)(x \otimes y) = \phi(\lambda, \mu)(x \otimes y)$.

Proof. Assume that P and Q are the orthogonal matrices such that

$$P^* A P = S, \quad Q^* B Q = T$$

where S and T are both upper triangular matrices. Note that $S \otimes T$ is still upper triangular and $P \otimes Q$ is still orthogonal, and

$$(P \otimes Q)^* (A^i \otimes B^j)(P \otimes Q) = (P^* A^i P) \otimes (Q^* B^j Q) = S^i \otimes T^j.$$

This identity implies that $(P \otimes Q)^* \phi(A, B)(P \otimes Q) = \phi(S, T)$. Since $\phi(S, T)$ is an upper triangular matrix whose diagonal entries consist of $\phi(\lambda_i, \mu_j)$, this proves that $\{\phi(\lambda_i, \mu_j) : \ i = 1 : m, j = 1 : n\}$ is the spectral set of $\phi(A, B)$. The rest of the proof is straighforward by definition. □

(1.5.7) Example. Consider the block triadiagonal matrix

$$A = \text{diag}\,(-I, B, -I) \quad \text{with} \quad B = \text{diag}\,(-1, 4, -1).$$

Obviously

$$A = I \otimes B + C \otimes I \quad \text{with} \quad C = \text{diag}\,(-1, 0, -1).$$

By Proposition 1.2.13, the eigenvalues of B and C are as follows:

$$\lambda_i(B) = 4 - 2\cos\frac{i\pi}{2(N+1)}, \quad \lambda_j(C) = -2\cos\frac{j\pi}{2(N+1)}.$$

By Theorem 1.5.6, the eigenvalues of A are given by

$$\begin{aligned} \lambda_{ij}(A) &= 4 - 2\cos\frac{i\pi}{N+1} - 2\cos\frac{j\pi}{N+1} \\ &= 4(\sin^2\frac{i\pi}{2(N+1)} + \sin^2\frac{j\pi}{2(N+1)}). \end{aligned}$$

Chapter 2

Basic Iterative Methods

Assume $\mathcal{V}$ is a finite dimensional vector space. The goal of this chapter is to study iterative methods and preconditioning techniques for solving the following kind of equation:

$$Au = f. \tag{2.0.1}$$

Here $A : \mathcal{V} \mapsto \mathcal{V}$ is an SPD linear operator over $\mathcal{V}$ and $f \in \mathcal{V}$ is given.

2.1 Elementary linear iterative methods

2.1-a Preliminaries

A single step linear iterative method which uses an old approximation, u^{old}, of the solution u of (2.0.1), to produce a new approximation, u^{new}, usually consists of three steps:

1. Form $r^{\text{old}} = f - Au^{\text{old}}$;
2. Solve $Ae = r^{\text{old}}$ approximately: $\hat{e} = Br^{\text{old}}$;
3. Update $u^{new} = u^{\text{old}} + \hat{e}$,

where B is a linear operator on $\mathcal{V}$ and it can be thought of as an approximate inverse of A.

As a result, we have the following

Algorithm 2.1.1 *Given $u^0 \in \mathcal{V}$,*

$$u^{k+1} = u^k + B(f - Au^k), \quad k = 0, 1, 2, \cdots ,$$

We say that an iterative scheme like (2.1.1) converges if $\lim_{k\to\infty} u_k = u$ for any $u_0 \in \mathcal{V}$.

The core of above iterate scheme is the operator B. Notice that if $B = A^{-1}$, after one iteration, u^1 is then the exact solution. In general, B may be regarded as an approximate inverse of A.

(2.1.2) Example. Assume $\mathcal{V} = \mathbb{R}^n$ and $A = (a_{ij}) \in \mathbb{R}^{n\times n}$ is the usual SPD matrix. We write $A = D - L - U$ with D being the diagonal of A and $-L$ and $-U$ the lower and upper triangular part of A respectively. The easiest approximate inverse of A are perhaps

$$B = D^{-1} \quad \text{or} \quad B = (D - L)^{-1}.$$

As we shall see that these two choices of B result in the well-known Jacobi and Gauss-Seidel methods.

Assume that u and u^k are solutions of (2.0.1) and (2.1.1) respectively, then

$$u - u^k = (I - BA)^k (u - u_0).$$

By (1.2.20), we have the following simple convergence result.

Lemma 2.1.3 *The iterative scheme* (2.1.1) *converges iff*

$$\rho(I - BA) < 1.$$

A trivial consequence (2.1.3) is that B is nonsingular SPD for a convergent scheme.

(2.1.4) Richardson iteration. Richardson iteration is perhaps the simplest iterative method which correspond to (2.1.1) with $B = \frac{\omega}{\rho(A)} I$. Namely,

$$u^{k+1} = u^k + \frac{\omega}{\rho(A)}(f - Au^k), \quad k = 0, 1, 2, \cdots, \tag{2.1.5}$$

An application of Lemma 2.1.3 shows that the Richardson iteration convergees if and only if $0 < \omega < 2$.

One can imagine that Richardson method is not very efficient method, but it is theoretically a very important method. One of the most remarkable property of this method is its "smoothing property":

(2.1.6) Symmetrization. Sometimes it is more desirable that the iterator B is symmetric. If B is not symmetric, there is a natural way to symmetrize it. Consider the following iteration

$$\begin{aligned} u^{k+1/2} &= u^k + B(f - Au^k) \\ u^{k+1} &= u^{k+1/2} + B^t(f - Au^{k+1/2}) \end{aligned}$$

where "t" denoting the transposition with respect to $(\cdot, \cdot)$. Eliminating the intermediate $u^{k+1/2}$ gives

$$u - u^{k+1} = (I - B^t A)(I - BA)(u - u^k)$$

or

$$u^{k+1} = u^k + \bar{B}(f - Au^k) \tag{2.1.7}$$

where, with "$*$" denoting the transposition with respect to $(\cdot,\cdot)_A$ and

$$\bar{B} = (I - (I - BA)^*(I - BA))A^{-1} = B^t + B - B^tAB \tag{2.1.8}$$

or

$$I - \bar{B}A = (I - BA)^*(I - BA). \tag{2.1.9}$$

Obviously $\bar{B}$ is symmetric and it is called the *symmetrization* of iterator B.

Lemma 2.1.10 *The following identities hold*

$$(\bar{B}Av, v)_A = ((2I - BA)v, BAv)_A \quad \forall\, v \in \mathcal{V}. \tag{2.1.11}$$

and

$$\|v\|_A^2 - \|(I - BA)v\|_A^2 = (\bar{B}Av, v)_A \quad \forall\, v \in \mathcal{V}.. \tag{2.1.12}$$

Theorem 2.1.13 *The following are equivalent:*

1. *The symmetrized scheme* (2.1.7) *is convergent.*
2. *The operator* $\bar{B}$ *given by* (2.1.8) *is SPD.*
3. *The matrix* $B^{-t} + B^{-1} - A$ *is SPD*
4. *There exists a constant* $\omega_1 \in (0,2)$ *such that*

$$(BAv, BAv)_A \le \omega_1 (BAv, v)_A \quad \forall\, v \in \mathcal{V}. \tag{2.1.14}$$

5. *There exists a constant* $\omega_1 \in (0,2)$ *such that*

$$(Av, v) \le \omega_1 (B^{-1}v, v) \quad \forall\, v \in \mathcal{V}. \tag{2.1.15}$$

6. *There exists a constant* $\omega_1 \in (0,2)$ *such that*

$$(\frac{2}{\omega_1} - 1)(Av, v) \le ((B^{-1} + B^{-t} - A)v, v) \quad \forall\, v \in \mathcal{V}. \tag{2.1.16}$$

7. *There exists a constant* $\omega_1 \in (0,2)$ *such that*

$$(2 - \omega_1)(Bv, v) \le (\bar{B}v, v) \quad \forall\, v \in \mathcal{V}. \tag{2.1.17}$$

Furthermore, the scheme 2.1.1 converges if (and only if, when B is symmetric) its symmetrized scheme 2.1.7 converges.

Proof. The proof for the equivalencies is straightforward by definition. By (2.1.9), $\rho(I - BA)^2 \le \rho(I - \bar{B}A)$, thus the scheme (2.1.1) converges if the symmetrized scheme (2.1.7) converges. If $B^t = B$ then, by (2.1.9) again, $\rho(I - BA)^2 = \rho(I - \bar{B}A)$ which implies that (2.1.1) converges iff (2.1.7) converges. $\Box$

Lemma 2.1.18
$$(2-\omega_1)B \le \bar{B} \le 2B.$$

(2.1.19) Richardson like iterative methods. As we shall see late, the smoothing property that the Richardson iteration has is very desirable in multigrid applications. We shall now discuss what kind of iterative method would satisfy a similar property.

An iterative method (2.1.1) is said to be Richardson like if there exists an $\omega \in (0,2)$ such that

$$\|(I-BA)v\|_A \le \|(I-\frac{\omega}{\rho_A}A)v\|_A \quad \forall\, v \in \mathcal{V}. \tag{2.1.20}$$

Lemma 2.1.21 *For the iterative method* (2.1.1)*, the followings are equivalent*

1. $(C_0\rho_A)^{-1}\|v\|^2 \le (\bar{B}v, v)$.
2. *The inequality* (2.1.20) *satisfies with* $\omega = C_0^{-1}$.
3. $(C_0\rho_A)^{-1}\|Av\|^2 \le \|v\|_A^2 - \|(I-BA)v\|_A^2$.

2.1-b Jacobi and Gauss-Seidel Methods

In the discussion of Jacobi and Gauss-Seidel method, we will confine ourselves to the case that A is the matrix in (2.0.1).

The simplest possible iterative method is perhaps the so-called *Richardson* iteratiion which is (2.1.1) by choosing $B = \omega I$ for some positive constant ω:

Algorithm 2.1.22 (Richardson) *For* $i = 1:n$

$$x_i^{k+1} = x_i^k + \omega\left(b_i - \sum_{j=1}^n a_{ij}x_j^k\right).$$

A slightly more complicated algorithm is the following *Jacobi* method.

Algorithm 2.1.23 (Jacobi) *For* $i = 1:n$

$$x_i^{k+1} = x_i^k + a_{ii}^{-1}\left(b_i - \sum_{j=1}^n a_{ij}x_j^k\right).$$

As we shall see that the Jacobi method may not be convergent for all SPD matrix. To fix this problem, a damping parameter can be introduced $B = \omega D^{-1}$.

Algorithm 2.1.24 (Damped Jacobi) *For* $i = 1:n$

$$x_i^{k+1} = x_i^k + \omega a_{ii}^{-1}\left(b_i - \sum_{j=1}^n a_{ij}x_j^k\right).$$

Gauss-Seidel method is obtained by revising the Jacobi method to use the most current estimate of the exact x_i, and it may also be obtained by taking $B = (D - L)^{-1}$.

Algorithm 2.1.25 (Forward Gauss-Seidel) *For* $i = 1 : n$

$$x_i^{k+1} = x_i^k + a_{ii}^{-1} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{k+1} - \sum_{j=i}^{n} a_{ij} x_j^k \right).$$

Algorithm 2.1.26 (Backward Gauss-Seidel) *For* $i = n : -1 : 1$

$$x_i^{k+1} = x_i^k + a_{ii}^{-1} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{k} - \sum_{j=i}^{n} a_{ij} x_j^{k+1} \right).$$

One way to improve the convergence of GS method is first to compute an intermidiate $\tilde{x}_i^{k+1}$ by (2.1.25) and then, for some constant ω, form

$$x_i^{k+1} = \omega \tilde{x}_i^{k+1} + (1 - \omega) x_i^k.$$

The resultant algorithm is called point relaxation method.

Algorithm 2.1.27 (SOR Method) *For* $i = 1 : n$

$$x_i^{k+1} = x_i^k + \omega a_{ii}^{-1} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{k+1} - \sum_{j=i}^{n} a_{ij} x_j^k \right).$$

We shall now derive more compact formulations for all above methods. For this purpose, we split the matrix in the form:

$$A = D - L - U$$

where D is the diagonal of A and $-L$ and $-U$ are the strictly lower and upper triangular parts where D is the diagonal of A and $-L$ and $-U$ are the strictly lower and upper triangular parts of A, respectively.

Equation $Ax = b$ then is equivalent to

$$Dx = (L + U)x + b.$$

The Jacobi, Gauss-Seidel and point relaxation methods can be written as

$$\begin{aligned} Dx^{k+1} &= (L + U)x^k + b, \\ Dx^{k+1} &= Lx^{k+1} + Ux^k + b, \\ Dx^{k+1} &= \omega L x^{k+1} + (\omega U + (1 - \omega)D)x^k + \omega b \end{aligned}$$

respectively.

We can write these methods in the form of (2.1.1) by some appropriate B. We have

$$B = \begin{cases} \omega & \text{Richardson;} \\ D^{-1} & \text{Jacobi;} \\ \omega D^{-1} & \text{Damped Jacobi;} \\ (D-L)^{-1} & \text{Gauss-Seidel;} \\ \omega(D-\omega L)^{-1} & \text{SOR.} \end{cases} \tag{2.1.28}$$

Theorem 2.1.29 *Assume A is SPD. Then*

- *Richardson method converges iff $0 < \omega < 2/\rho(A)$;*
- *Jacobi method converges iff $2D - A$ is SPD;*
- *Damped Jacobi method converges iff $0 < \omega < 2/\rho(D^{-1}A)$;*
- *Gauss-Seidel method always converges;*
- *SOR method converges iff $0 < \omega < 2$.*

Proof. By (2.1.28), we have

$$B^{-t} + B^{-1} - A = \begin{cases} 2/\omega - A & \text{Richardson;} \\ 2D - A & \text{Jacobi;} \\ 2D/\omega - A & \text{Damped Jacobi;} \\ D & \text{Gauss-Seidel;} \\ \frac{2-\omega}{\omega} D & \text{SOR.} \end{cases}$$

Therefore all the desired results follow from (2.1.13). □

(2.1.30) EXERCISE. Prove that the condition that $0 < \omega < 2$ is also necessary for the convergence of SOR method.

The symmetrized version of GS and SOR methods are often useful. As GS is a special case of SOR, we shall now only discuss the symmtric SOR, known as SSOR. The algorithm is as follows:

Algorithm 2.1.31 (SSOR Algorithm)

$$\begin{aligned} Dx^{k+\frac{1}{2}} &= \omega L x^{k+\frac{1}{2}} + (\omega U + (1-\omega)D)x^k + b, \\ Dx^{k+1} &= \omega U x^{k+1} + (\omega L + (1-\omega)D)x^{k+\frac{1}{2}} + b. \end{aligned}$$

In this case, it is easy to check that the iterator is

$$B = \omega(2-\omega)(D-\omega U)^{-1} D (D-\omega L)^{-1}$$

which is SPD if $\omega \in (0, 2)$. A similar argument shows that SSOR converges iff $\omega \in (0, 2)$.

Theorem 2.1.32 *Assume that A is an SPD block tri-diagonal matrix, then the Jacobi method always converges, namely $\rho(I-D^{-1}A)<1$. Furthermore, the optimal relaxation parameter ω is given by*

$$\omega_{\rm opt} = \frac{2}{1+\sqrt{1-\rho(I-D^{-1}A)}}$$

for which $\rho(I-\omega(D-\omega L)^{-1}A)$ is minimized with respect to $\omega \in (0,2)$.

(2.1.33) Kaczmaz method. If the operator A in (2.0.1) is not SPD (but nonsingular, of course), consider the normal equation of (2.0.1) as follows:

$$A^tAu = A^tf. \tag{2.1.34}$$

Since A^tA is SPD, the above problem can then be solved by, for example, congugate gradient and GS methods.

Even though this approach works in principle, but it is often not recommended. The main reason is that the condition number of A^tA will be like the square of that of A, hence the nomal equation is usually quite ill-conditioned.

Nevertheless, the Gauss-Seidel method for the (2.1.34) is not unpopular. The resultant algorithm, known as Kaczmarz method, is as follows:

$$x_i^{k+1} = x_i^k + a_i^t[b_i - \sum_{j=1}^{i-1} a_j x_j^{k+1} - \sum_{j=i}^{n} a_j x_j^k]/a_i^t a_i$$

where $a_i \in \mathbb{R}^n$ is the i-th column of A.

Another variant of Kaczmarz method can be obtained considering the following relation

$$AA^tv = r, \quad u = A^tv.$$

2.1-c Linear equations in terms of stiffness matrices

If $\mathcal{V}$ is not $\mathbb{R}^n$, the linear equation (2.0.1) needs to be translated into a system of equations in terms of matrices and vectors in $\mathbb{R}^n$.

To this end, we note that the equation (2.0.1) is equivalent to

$$(Au, v) = (f, v) \quad \forall\, v \in \mathcal{V}.$$

Given a absis (ϕ_i) of $\mathcal{V}$, write $u = \sum_j \mu_j$, we then have

$$\sum_j (\phi_i, A\phi_j)\mu_j = (f, \phi_i), \quad i = 1:N.$$

We have therefore reached the following linear algebraic system

$$\mathcal{A}\mu = \eta. \tag{2.1.35}$$

Similar to (2.1.1), a linear iterative method for (2.1.35) can be written as

$$\mu^{k+1} = \mu^k + \mathcal{B}(\eta - \mathcal{A}\mu^k), \quad k = 0, 1, 2, \cdots, \tag{2.1.36}$$

where $\mathcal{B} \in \mathbb{R}^{n \times n}$.

Proposition 2.1.37 *u is the solution of* (2.0.1) *if and only if $\mu = \widetilde{u}$ is the solution of* (2.1.35) *with $\eta = \mathcal{M}\widetilde{f}$. The linear iterations* (2.1.1) *and* (2.1.36) *are equivalent if and only if $\widetilde{B} = \mathcal{B}\mathcal{M}$. In this case, $\sigma(\mathcal{B}\mathcal{A}) = \sigma(BA)$ and hence $\kappa(\mathcal{B}\mathcal{A}) = \kappa(BA)$.*

Proof. If u is the solution of (2.0.1), by (**??**), $\widetilde{A}\widetilde{u} = \widetilde{Au} = \widetilde{f}$. But, by (**??**), $\widetilde{A} = \mathcal{M}^{-1}\mathcal{A}$ and hence $\mathcal{A}\widetilde{u} = \mathcal{M}\widetilde{f}$. Furthermore, $\sigma(BA) = \sigma(\widetilde{BA}) = \sigma(\mathcal{B}\mathcal{A})$. □

2.1-d Alternative formulations of iterative schemes

Lemma 2.1.38 *The Gram matrix $\mathcal{M}$ is the matrix representation of the linear operator $R \in L(\mathcal{V})$ given by*

$$Rv = \sum_{i=1}^{n} (v, \phi_i)\phi_i \quad \forall\, v \in \mathcal{V}. \tag{2.1.39}$$

(2.1.40) EXERCISE. Prove Lemma (2.1.38).◇

Proposition 2.1.41 *The Richardson iteration for* (2.1.35) *is equivalent to the iteration for* (2.0.1) *with $B = \omega R$, where R is defined by* (2.1.39).

Proof. The iteration for (2.0.1) with $B = \omega R$ is

$$u^{k+1} = u^m + \omega R(f - Au^k).$$

It follows from (**??**) and (**??**) that

$$\begin{aligned} \mu^{k+1} = \widetilde{u^{k+1}} &= \widetilde{u^k} + \omega\widetilde{R}(\widetilde{f} - \widetilde{A}\widetilde{u^k}) \\ &= \mu^k + \omega\mathcal{M}(\widetilde{f} - \mathcal{M}^{-1}\mathcal{A}\mu^k) \\ &= \mu^k + \omega(\eta - \mathcal{A}\mu^k) \end{aligned}$$

This is exactly the Richardson iteration for (2.1.35). □

A similar argument leads to the following conclusion.

Proposition 2.1.42 *The Damped Jacobi iteration for* (2.1.35) *is equivalent to the iteration for* (2.0.1) *with B given by*

$$Bv = \omega \sum_{i=1}^{n} a_{ii}^{-1}(v, \phi_i)\phi_i \quad \forall\, v \in \mathcal{V}.$$

(2.1.43) EXERCISE. Derive the abstract formulation for Gauss-Seidel method.

Chapter 3

An Introduction to Multigrid Method Through A One Dimensional Example

3.1 Basic multigrid ideas

In this section, we discuss some multigrid ideas by studying a simple two point boundary value problem and its finite difference discretizations. The presentation is informal but hopefully informative. Numerical examples are used to illustrate the main ideas.

The prerequisite for this section is some basic linear algebra and calculus.

3.1-a Two point boundary value problem and finite difference discretization

We consider the following simplest possible elliptic boundary problem:

$$-u'' = f, \quad x \in (0,1) \qquad u(0) = u(1) = 0. \tag{3.1.1}$$

We use a simple central finite difference method to discretize the above equation. To do that, for any integer N, we consider a uniform grid, denoted by $\mathcal{T}_h$, of the interval $[0,1]$ as follows:

$$0 = x_0 < x_1 < \cdots < x_{N+1} = 1, \quad x_j = \frac{j}{N+1}(j = 0 : N+1). \tag{3.1.2}$$

This partition consists of uniform subintervals of the length $h = \frac{1}{N+1}$.

Since a derivative is a limit of some difference quotient, we can employ

$$u''(x_j) \approx \frac{u'(x_{j+1/2}) - u'(x_{j-1/2})}{h},$$

where $x_{j+1/2} = (x_{j+1} + x_j)/2$. Similarly

$$u'(x_{j+1/2}) \approx \frac{u(x_{j+1}) - u(x_j)}{h},$$

$$u'(x_{j-1/2}) \approx \frac{u(x_j) - u(x_{j-1})}{h}.$$

Thus for a smooth solution u, we have

$$u''(x_j) \approx \frac{u(x_{j-1}) - 2u(x_j) + u(x_{j+1})}{h^2}.$$

Therefore, if $\mu = (\mu_i) \in R^N$ satisfies

$$\frac{-\mu_{j-1} + 2\mu_j - \mu_{j+1}}{h^2} = f(x_j), \quad 1 \le j \le N, \quad \mu_0 = \mu_{N+1} = 0,$$

then

$$\mu_j \approx u(x_j).$$

It is natural to expect that this approximation gets more accurate as N gets larger.

The finite difference discretization gives rise to a discrete set of approximation (μ_j). It is often convenient to interpolant these discrete values into some function that is defined everywhere on the interval $[0, 1]$. One natural way is the *linear interpolation*. The linear interpolation of the vector (μ_j) is a piecewise linear function, u_h, with respect to the partition (3.1.2), such that

$$u_h(x_j) = \mu_j, \quad 0 \le j \le N+1, \quad \mu_0 = \mu_{N+1} = 0.$$

Figure 3.1 shows the plots of the exact solution of (3.1.1) with $f(x) = 1$ and linearly interpolated u_h from its finite difference approximation with $N = 5$ and $N = 7$ and $N = 15$ respectively. As we see the approximation property of the finite difference discretization is reasonable.

3.1-b Algebraic system for the discrete solution. Richardson iteration

Our main concern here is on how to effectively find the discrete solution, namely to solve the following linear algebraic system for μ:

(3.1.3) $$A\mu = \beta.$$

where

$$A = \frac{1}{h}\mathrm{diag}(-1, 2, -1), \beta_j = hf(x_j).$$

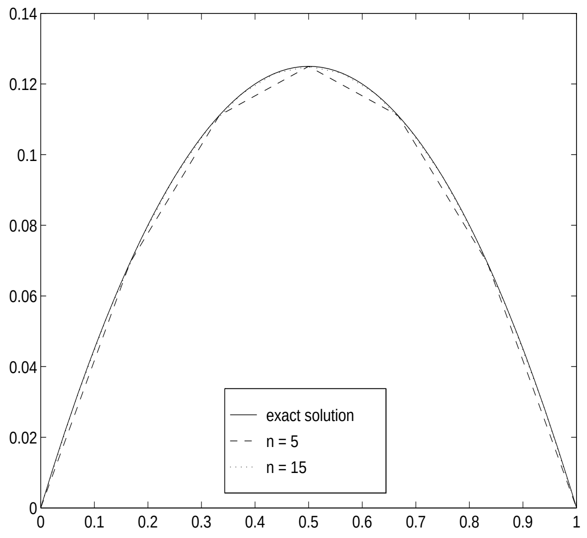


Figure 3.1: An exact solution and two approximate solutions

Here $\mathrm{diag}(-1, 2, -1)$ denotes the tridiagonal matrix with main diagonal entries being 2 and both sub-diagonals being -1.

One probably wonder why we choose to keep a scaling factor of $1/h$ in the above definition of stiffness matrix. The main reason to do so is to make our notation be consistent with that from the finite element method we will discuss more late. Although this additional constant factor may look a little awkward, it should not add much complication.

Of course, this particular tridiagonal system can be easily solved by many methods such as banded Gaussian elimination. But for illustrational purpose, we shall instead consider to use iterative methods.

One very simple iterative method for (3.1.3) is the following Richardson method

$$\mu^l = \mu^{l-1} + \sigma^{-1}(b - A\mu^{l-1}),$$

or, for $j = 1 : N$,

$$\mu_j^l = \mu_j^{l-1} + \sigma^{-1}\left(\beta_j - \frac{-\mu_{j-1}^{l-1} + 2\mu_j^{l-1} - \mu_{j+1}^{l-1}}{h}\right),$$

where $\sigma > 0$ is a positive parameter.

It is not so difficult to properly choose σ so that the above iterative scheme converges, namely for any initial guess μ^0, the sequence (μ^l) generated by the above iteration converges to the exact solution μ of (3.1.3):

Note that

$$\mu = \mu + \sigma^{-1}(b - A\mu),$$

we get

$$\mu - \mu^l = (I - \sigma^{-1}A)(\mu - \mu^{l-1}),$$

or

$$\mu - \mu^l = (I - \sigma^{-1}A)^l(\mu - \mu^0), l = 1, 2, 3, \cdots$$

As we known,

$$(I - \sigma^{-1}A)^l \longrightarrow o$$

if and only if $\rho(I - \sigma^{-1}A) < 1$. Here $\rho(B)$ is the spectral radius of matrix B. However, $\rho(I - \sigma^{-1}A) < 1$ if and only if

$$0 < \text{all the eigenvalue of } A < 2\sigma.$$

Thus, a necessary and sufficient condition for the convergence is the following

$$\sigma > \rho(A)/2.$$

It is easy to see that (for example, $4/h$ is an upper bound of its raw sums)

$$\rho(A) < 4/h.$$

Therefore it is reasonable to make the following choice:

$$\sigma = 4/h$$

and the resulting algorithm is

$$\mu^l = \mu^{l-1} + \frac{h}{4}(b - A\mu^{l-1}). \tag{3.1.4}$$

In the rest of this section, unless otherwise noted, we shall choose σ as above for simplicity.

On Figure 3.2 the convergence history plot of the above Richardson iterative method for typical application is shown. As we see, this iterative scheme converges very slowly.

Our main goal is to find a way to speed up such kind of rather slowly convergent iterative scheme. To do that, we need to study its convergent property in more microscopic level. First of all, let us now take a careful look at the convergence history picture and make the following observation:

> *Observation 1.* The scheme converges rather fast in the very beginning but then slows down after a few step.

To further understand this phenomenon, let us plot the detailed pictures of the error functions in the first few iterations. After a careful look at these pictures, we have the following observation:

> *Observation 2.* The scheme not only converges fast in the first few steps, but also smooth out the error function very quickly.

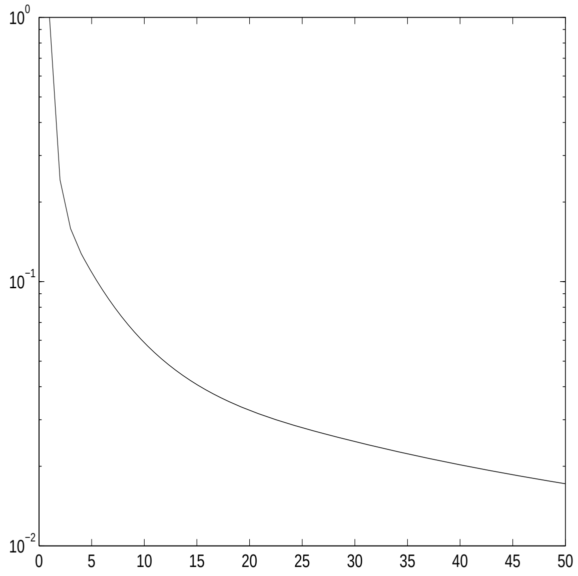


Figure 3.2: A picture on the Richardson method convergence history

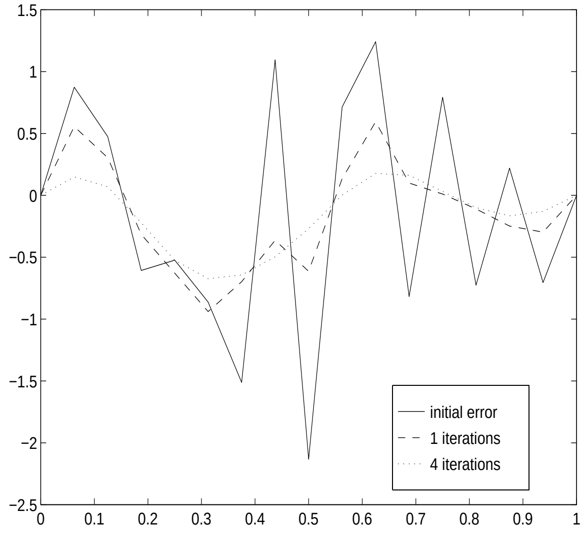


Figure 3.3: The smoothing effect of the Richardson method

In other words, the error function becomes a much smoother function after a few such simple iterations. This property of the iterative scheme is naturally called *a smoothing property* and an iterative scheme having this smoothing property is called a *smoother*.

The above two observations, especially the second one, concern the most important property of the simple Richardson method that we can take advantage to get a much faster algorithm.

3.1-c Coarse grid correction and a two-grid method

As we discussed above, although the Richardson iteration does not converges very fast, it does quickly smooth out the rough component in the error. We shall now explore how to speed up the convergence. The main idea is to use the fact that a relatively smooth function can be well approximated on a relatively coarse grid. Now, since the error is smooth after a few Richardson iterations, we should be able to correct the residual effectively using a coarser grid.

Let μ^m denote the approximate solution obtained after m-th iteration of the Richardson method and $r = b - A\mu^m$. We consider the residual equation as follows:

$$A_h \epsilon^h = \gamma^h. \tag{3.1.5}$$

Here $A_h = A$ and $\gamma^h = \gamma$ and we use the superscript h to refer that the above equation is on the grid of size h. Here $\epsilon = \mu - \mu^m$. Since ϵ is smooth, ϵ should be approximated well on a grid coarser than the original grid. For simplicity, we assume that N is an odd integer and we take the coarse grid, denoted by $\mathcal{T}_{2h}$ to be a grid of size $2h$ with the following grid points

$$x_i^{2h} = x_{2i}, i = 1 : N_{2h} \equiv (N+1)/2. \tag{3.1.6}$$

We would like to restrict the equation (3.1.5) to the above coarse grid and then solve the coarse grid equation. To do this, we first need to find a reasonable way for obtaining the restriction of (3.1.5) to the coarse grid. To this end, we can imagine that (3.1.5) is the linear system corresponding to a finite different discretization of a two point boundary value problem:

$$-e'' = g, \quad x \in (0,1) \qquad e(0) = e(1) = 0. \tag{3.1.7}$$

Here g is some function satisfying

$$\gamma_i = hg(x_i). \tag{3.1.8}$$

Now, we use finite difference to discretize (3.1.7) on the coarse grid $\mathcal{T}_{2h}$ and obtain the corresponding system as follows:

$$A_{2h}\epsilon^{2h} = \gamma^{2h}, \tag{3.1.9}$$

where $A_{2h} = (2h)^{-1}\text{diag }(-1, 2, -1) \in R^{N_{2h}\times N_{2h}}$ is the stiffness matrix from the coarse grid $\mathcal{T}_{2h}$ and

$$\gamma_i^{2h} = 2hg(x_i^{2h}). \tag{3.1.10}$$

Comparing (3.1.8) with (3.1.10) and using (3.1.6), we obtain the following relation for the components of the restricted vector γ^{2h}:

$$\gamma_i^{2h} = 2\gamma_{2i}^h. \tag{3.1.11}$$

The equation (3.1.9) with (3.1.11) is a desired restricted equation of (3.1.5) to the coarse grid $\mathcal{T}_{2h}$.

We would like to mention the following slightly different restriction is often used instead of (3.1.11):

$$\gamma_i^{2h} = \frac{1}{2}\gamma_{2i-1}^h + \gamma_{2i}^h + \frac{1}{2}\gamma_{2i+1}^h, \tag{3.1.12}$$

which is apparently very close to (3.1.11) since, for a smooth residual, $\gamma_{2i-1}^h \approx \gamma_{2i+1}^h \approx \gamma_{2i}^h$. We can imagine that using an averaged value should be better than a single value. We shall see in the next section, this is the most natural restriction in the finite element setting. This restriction also has a more interesting relation with the prolongation matrix that we will introduce in a moment.

The relation (3.1.11) or (3.1.12) defines a so-called restriction matrix $I_h^{2h} : R^{N_h} \mapsto R^{N_{2h}}$ such that

$$\gamma_{2h} = I_h^{2h}\gamma^h.$$

The equation (3.1.9) is half the size of (3.1.5). Let us solve this equation exactly and obtain:

$$\epsilon^{2h} = A_{2h}^{-1}\gamma^{2h}.$$

This ϵ^{2h} is the coarse grid correction of μ^m. To add this correction to μ^m, we need to think ϵ^{2h} to be the nodal value vector of a piecewise linear function defined on $\mathcal{T}_{2h}$. Apparently this piecewise linear function can be evaluated at the grid points of $\mathcal{T}_h$ as follows:

$$\tilde{\epsilon}_{2i}^h = \epsilon_i^{2h}, \tilde{\epsilon}_{2i+1}^h = \frac{1}{2}(\epsilon_i^{2h} + \epsilon_{i+1}^{2h}), \quad i = 1 : N_{2h}. \tag{3.1.13}$$

The above relation defines a so-called prolongation matrix $I_{2h}^h : R^{N_{2h}} \mapsto R^{N_h}$ such that

$$\tilde{\epsilon}^h = I_{2h}^h \epsilon^{2h}.$$

It is easy to see this prolongation matrix is related to the restriction matrix defined by (3.1.12) by the following identity:

$$I_{2h}^h = [I_h^{2h}]^t,$$

where t is the transposition. Because of this relationship, the restriction given by (3.1.12) is particularly interesting.

With this prolongation matrix, the approximation μ^m can be updated as follows

$$\mu^m + I_{2h}^h \epsilon^{2h}.$$

Algorithm 3.1.14 (A two grid method) *Given μ^0, $\mu \leftarrow \mu^0$.*

1. *Fine grid smoothing: applying m times Richardson iterations to obtain μ^m.*
2. *Coarse grid correction: solving the residual equation restricted on the coarse grid $\mathcal{T}_{2h}$ to obtain*
$$\epsilon^{2h} = A_{2h}^{-1}(I_h^{2h}\gamma^h).$$
3. *Update: $\mu \leftarrow \mu^m + I_{2h}^h\epsilon^{2h}$.*
4. *Stop if convergent or continue from step 1.*

Let us now give a numerical example to show how well the above two-grid method works.

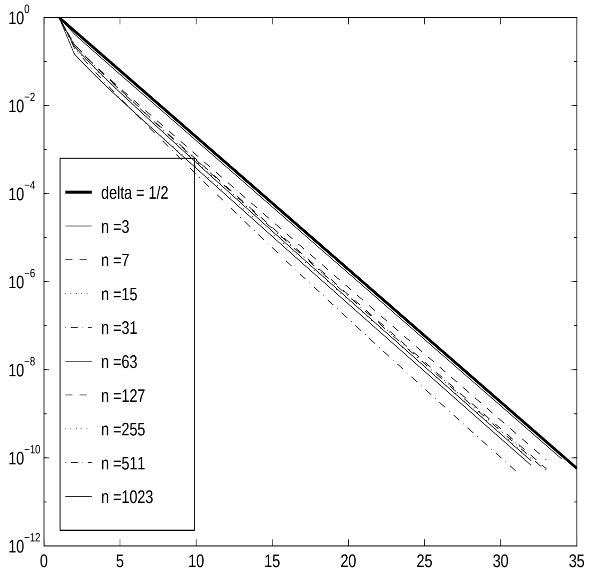


Figure 3.4: Convergence history of the two level method for different number of unknowns (different values of h respectively). The thick line corresponds to damping factor 1/2. Only one smoothing step is applied in the algorithm.

A different formulation and possible multigrid extension We would like to present the above two-grid method in a slightly different but equivalent fashion. Let us first only consider the above algorithm in the following special case:
$$\mu^0 = 0, \text{with step 4 removed}$$
This means nothing but one sweep of Algorithm 3.1.14 with zero initial guess. Recall we are trying to solve the system $A_h\xi^h = \beta^h$. Given the right hand side

$\beta^h \in R^{N_h}$, with the aforementioned one sweep of two-grid iteration, we obtain an approximate solution which we denote by $\tilde{\xi}^h \in R^{N_h}$. This procedure defines the following transformation:

$$\beta^h \mapsto \tilde{\xi}^h.$$

It is not hard to see that the above transformation is in fact linear and hence it corresponds to a matrix $B_h \in R^{N_h \times N_h}$. Namely

$$\tilde{\xi}^h = B_h \beta^h.$$

Since $\tilde{\xi}^h$ is meant to be an approximation of the solution of the equation $A_h \xi^h = \beta^h$, B_h can then thought as an approximate inverse of A_h.

For clarity, let us reiterate the detailed definition of B_h as follows:

Algorithm 3.1.15 (An algorithm defining the action of B_h) *Let $\beta^h \in R^{N_h}$ be given.*

1. *Fine grid smoothing: with $\mu^0 = 0$,*

$$\mu^l = \mu^{l-1} + \sigma^{-1}(b - A\mu^{l-1}),\ l = 1 : m.$$

2. *Coarse grid correction: Form the residual $\gamma^h = \beta^h - A_h \mu^m$ and solve the coarse grid residual equation $A_{2h}\epsilon = I_h^{2h}\gamma^h$ exactly*

$$\epsilon^{2h} = A_{2h}^{-1}(I_h^{2h}\gamma^h).$$

3. *Define: $B_h \beta^h = \mu^m + I_{2h}^h \epsilon^{2h}$.*

With the B_h defined above, it is clear that Algorithm 3.1.15 is equivalent to the following iteration:

$$\mu \leftarrow \mu^0, \text{ iterate } \mu \leftarrow \mu + B_h(\beta^h - A_h\mu) \text{ till convergence.}$$

In the above two-grid algorithm, an exact coarse grid solver on $\mathcal{T}_{2h}$ is used. If the grid $\mathcal{T}_{2h}$ is not very coarse, such an exact solver can still be very costly. To improve the efficiency, we need to replace this exact solver by some appropriate approximate solver. The idea is to repeat the same two-grid method on this equation on $\mathcal{T}_{2h}$ by using a further coarse grid, say $\mathcal{T}_{4h}$. We continue this process till we reach a very coarse mesh for which an exact solver can be easily used. The resulting algorithm is a typical multigrid algorithm which we shall discuss in more details in the next section.

3.1-d Multilevel coarse grid corrections and a multigrid method

To describe a multigrid algorithm, we first need to have a multiple level of grids, say $\mathcal{T}_k$ with $k = 1 : J$ and $\mathcal{T}_J = \mathcal{T}_h$ being the finest mesh. There are many ways to obtain multiple level of grids and one simple definition of the grid points in $\mathcal{T}_k$ is as follows:

$$x_i^k = \frac{i}{2^k}, \quad i = 1, 2, \cdots, N_k, k = 1, 2, \cdots, J,$$

where $N_k = 2^k - 1$. Note that $\mathcal{T}_k$ can be viewed as being obtained by adding midpoints of the subintervals in $\mathcal{T}_{k-1}$. For each k the set of above nodes will be denoted by $\mathcal{N}_k$.

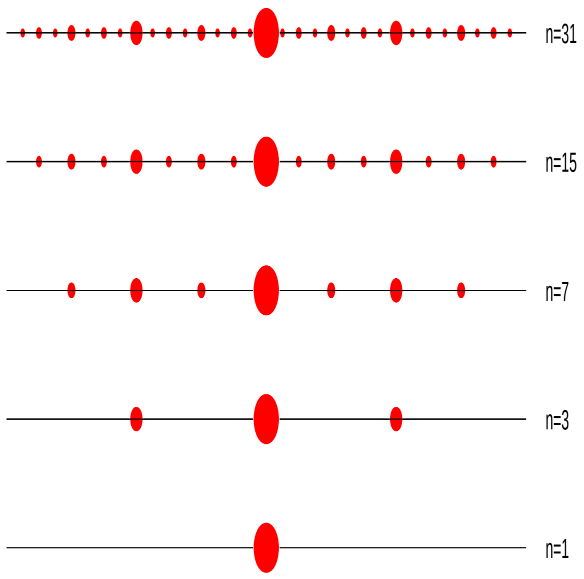


Figure 3.5: Multiple grids in one dimension

With our previous experiences in two-grid method, the description of a multi-grid method is not very difficult. In fact, the multiple level grids are treated by treating each two consecutive grids, say $\mathcal{T}_k$ versus $\mathcal{T}_{k-1}$. If we think $\mathcal{T}_k$ versus $\mathcal{T}_{k-1}$ like $\mathcal{T}_h$ and $\mathcal{T}_{2h}$, then there is not much new in the multigrid setting.

Let us give some details the definition of the restriction and prolongation matrices. The restriction matrix $I_k^{k-1} : R^{N_k} \mapsto R^{N_{k-1}}$ can be defined, as in (3.1.12), by

$$\gamma^{k-1} = I_k^{k-1}\gamma^k : \gamma_i^{k-1} = \frac{1}{2}\gamma_{2i-1}^k + \gamma_{2i}^k + \frac{1}{2}\gamma_{2i+1}^k. \tag{3.1.16}$$

The prolongation matrix $I_{k-1}^k : R^{N_{k-1}} \mapsto R^{N_k}$ can be defined as in (3.1.13) by

$$\epsilon^k = I_{k-1}^k\epsilon^{k-1} : \epsilon_{2i}^k = \epsilon_i^{k-1}, \epsilon_{2i+1}^k = \frac{1}{2}(\epsilon_i^{k-1} + \epsilon_{i+1}^{k-1}), \; i = 1 : N_{k-1}. \tag{3.1.17}$$

With the restriction and prolongation matrices in hands, we can now present a multilevel version of the earlier two-grid algorithm. As mentioned before, the idea is to repeat this two grid process for the coarse grid by using an even coarser grid. The resulting algorithm is just a desired multigrid algorithm. It is not hard to see that this kind of algorithm can be defined recursively.

Like the matrix B_h introduced in Algorithm 3.1.15, we would like to present the multigrid method by defining a sequence of matrices $B_k : R^{N_k} \mapsto R^{N_k}$ which is an approximate of A_k.

Algorithm 3.1.18 (A multigrid algorithm) *Define $B_1 = A_1^{-1}$. Assume $B_{k-1} : R^{N_{k-1}} \mapsto R^{N_{k-1}}$ is defined. To define $B_k : R^{N_k} \mapsto R^{N_k}$, we consider to approximately solve the equation $A_k\mu = \beta^k$ for $\beta^k \in R^{N_k}$.*

1. *Fine grid smoothing: with $\mu^0 = 0$,*
$$\mu^l = \mu^{l-1} + \sigma_k^{-1}(\beta^k - A_k\mu^{l-1}),\ l = 1 : m.$$
2. *Coarse grid correction: form the residual $\gamma^k = \beta^k - A_k\mu^m$ and solve the coarse grid residual equation $A_{k-1}\epsilon = I_k^{k-1}\gamma^k$ by using B_{k-1}*
$$\epsilon^{k-1} = B_{k-1}I_k^{k-1}\gamma^k.$$
3. *Define: $B_k\beta^k = \mu^m + I_{k-1}^k\epsilon^{k-1}$.*

The above algorithm defines B_k by using B_{k-1} recursively. The following is another procedure for producing B_k.

Algorithm 3.1.19 (An equivalent implementation) *Given $\beta \in R^{N_J}$, we need to compute $B_J\beta$(m=1).*

```
$\beta_J = \beta$;
$for$    $k = J : 2$,
         $\alpha_k = \sigma_k^{-1}\beta_k$,
         $\beta_{k-1} = I_k^{k-1}(\beta_k - A_k\alpha_k)$;
$endfor$
         $\alpha_1 = A_1^{-1}\beta_1$,
$for$    $k = 2 : J$,
         $\alpha_k \longleftarrow \alpha_k + I_{k-1}^k\alpha_{k-1}$;
$endfor$
$\beta_J\beta = \alpha_J$.
```

With the B_k defined above, a typical multigrid iteration can be obtained.

Algorithm 3.1.20

$$\mu \leftarrow \mu^0,\ \textit{iterate}\ \mu \leftarrow \mu + B_J(\beta^J - A_J\mu)\ \textit{till convergence},$$

where the action of B_J is computed by Algorithm 3.1.18.

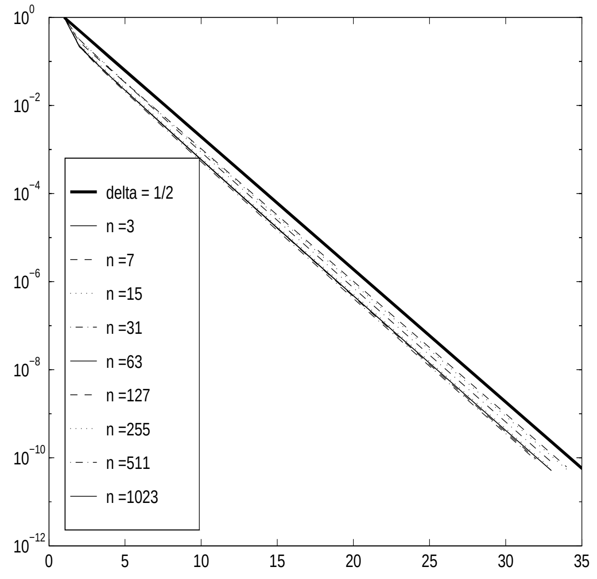


Figure 3.6: Convergence history of multilevel method for different number of unknowns (different values of h respectively). The thick line corresponds to damping factor 1/2. Only one smoothing step is applied in the algorithm.

Let us now give a numerical example to demonstrate the efficiency of the above multigrid method.

As we can see from the numerical examples, the multigrid algorithm converges very fast. In fact, its convergence rate is independent of the grid size. We will prove in the next section that there exists a constant $\delta \in (0, 1)$ independent of h such that

$$\|\mu - \mu^k\|_{A_J} \leq \delta^k \|\mu - \mu^0\|_{A_J},$$

where μ^k is the $k - th$ iteration by the multigrid Algorithm 3.1.18.

3.1-e Summary

In this section, we have tried to explain the main ideas of multigrid methodology by using a very simple two point boundary value problem with the simple finite difference discretization. Despite its simplicity, this model reflects most of the main features of more general and more complicated elliptic boundary value problems.

Roughly speaking, a multigrid method is based on two important observations. First, a local relaxation method such as Richardson iteration can damp out non-smooth part of the error and after a few Richardson iterations, the residual become a relatively smooth vector. Secondly, a relatively smooth vector can be well approximated by a relatively coarser grid. Combining the above two observations, a local relaxation iterative scheme can then be speeded up by correcting the residual on coarse grid. Applying this idea recursively with a multiple level of grids, an efficient multigrid method is then obtained.

3.2 Why multigrid works?

In this section we give some simple mathematical analysis of some aspects of the multigrid method introduced in the previous section. This analysis should be helpful to have a better understanding why the multigrid idea works.

Again the prerequisites for this section is basic linear algebra and one variable calculus.

3.2-a Why Richardson iteration has smoothing property?

Because of the extraordinary importance of this smoothing property, we shall now try to give some simple theoretical analysis. To do this, we make use of the eigenvalues and eigenvectors of the matrix A.

We recall that λ is an eigenvalue of A and and $\xi \in R^N \backslash \{0\}$ is a corresponding eigenvector if

$$A\xi = \lambda\xi.$$

Because of the special structure of A, all the N eigenvalues, λ_k, and the corresponding eigenvectors, $\xi^k = (\xi^k_j)$, of A can be obtained, for $1 \le k \le N$, as follows:

$$\lambda_k = \frac{4}{h}\sin^2\frac{k\pi}{2(N+1)}, \xi^k_j = \sin\frac{kj\pi}{N+1}(1 \le j \le N).$$

Indeed, the relation $A\xi^k = \lambda_k\xi^k$ can be verified by following elementary trigonometric identities:

$$-\sin\frac{k(j-1)\pi}{N+1} + 2\sin\frac{kj\pi}{N+1} - \sin\frac{k(j+1)\pi}{N+1} = 4\sin^2\frac{k\pi}{2(N+1)}\sin\frac{kj\pi}{N+1}.$$

Actually, it is not very difficult to derive these formula directly (see appendix A).

To understand the behavior of these eigenvectors, let us take $N = 6$ and plot the linear interpolations of all these 6 eigenvectors. We immediately observe that each vector ξ^k corresponds to a given frequency, and larger k corresponds to higher frequency. As λ_k is increasing with respect to k, we can then say that a larger eigenvalue of A corresponds to a higher frequency eigenvector. From a numerical point of view, we say a relatively low frequency vector is relatively smoother whereas a high frequency vector is nonsmooth.

We note that the set of eigenvectors $\xi^k = (\xi^k_j)$ forms an orthogonal basis of R^N. (This fact can be checked directly, or it also follows from the fact that the matrix A is symmetric and has N distinctive eigenvalues.) Therefore, any

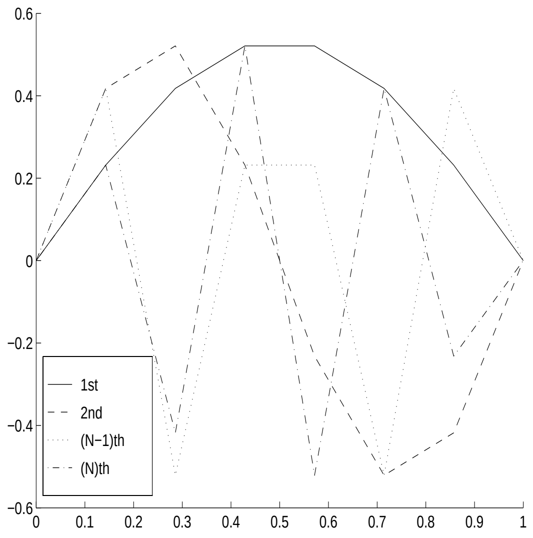


Figure 3.7: The eigenvectors

$\xi \in R^N$ can be expanded in terms of these eigenvectors:

$$\xi = \sum_{k=1}^{N} \alpha_k \xi^k.$$

This type of expansion is often called discrete Fourier expansion. The smoothness of the vector ξ has a lot to do with the relative size of the coefficients α_k. To see this numerically, let us again take $N = 4$ and consider the following two vectors

$$\xi = \sum_{k=1}^{4} 2^{1-k} \xi^k, \quad \eta = \sum_{k=1}^{4} 2^{k-4} \xi^k.$$

The first vector ξ has larger coefficients in front of lower frequencies whereas the second vector η has larger coefficients in front of higher frequencies. From Figure 3.8, it is easy to see how the smoothness of a vector depends on the relative size of its Fourier coefficients. We conclude that, in general, a vector with relatively small Fourier coefficients in front of the higher frequencies is relatively smooth and conversely, a vector with relatively large Fourier coefficients in front of the higher frequencies is relatively rough or nonsmooth.

Fourier analysis for the Richardson method Our earlier numerical experiments indicate that the Richardson method has a smoothing property. Based on our understanding of the relation between the smoothness and the size of

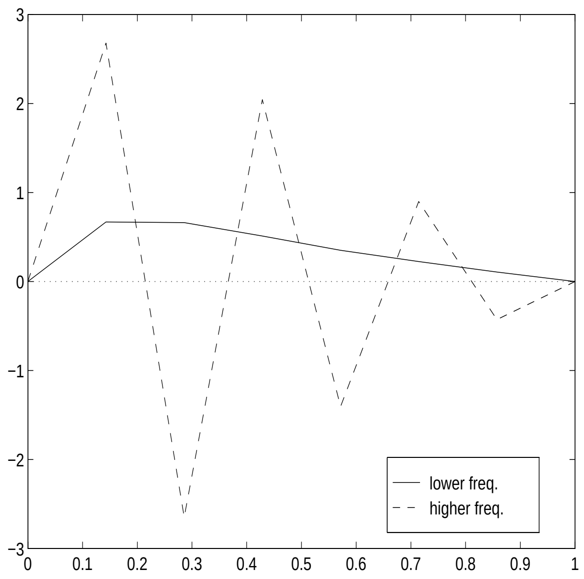


Figure 3.8: Plots of ξ and η. ξ-solid line; η dashed line

Fourier coefficients, we can imagine that this smoothing property can be analyzed using the discrete Fourier expansion.

Let μ be the exact solution of (3.1.3) and μ^l the result of $l-th$ iteration from the damped Jacobi (3.1.4). Then

$$\mu - \mu^l = (1 - \sigma^{-1}A)(\mu - \mu^{l-1}) = \ldots = (1 - \sigma^{-1}A)^l(\mu - \mu^0).$$

Consider the Fourier expansion of the initial error:

$$\mu - \mu^0 = \sum_{k=1}^{N} \alpha_k \xi^k.$$

Then

$$\mu - \mu^l = \sum_{k=1}^{N} \alpha_k (I - \sigma^{-1}A)^l \xi^k.$$

Note that $\sigma = 4/h$ and for any polynormail p

$$p(A)\xi^k = p(\lambda_k)\xi^k,$$

we get

$$\mu - \mu^m = \sum_{k=1}^{N}(1 - \lambda_k/\sigma)^m \xi^k = \sum_{k=1}^{N} \alpha_k^m \xi^k$$

where

$$\alpha_k^m = \left(1 - \sin^2 \frac{k\pi}{2(N+1)}\right)^m \alpha_k.$$

Note that

$$1 - \sin^2 \frac{k\pi}{2(N+1)} = \cos^2 \frac{k\pi}{2(N+1)}$$

implies

$$\alpha_k^m = \alpha_k \sin^{2m} \frac{N-k+1}{N+1}\frac{\pi}{2} \le \alpha_k \left(\frac{N-k+1}{N+1}\frac{\pi}{2}\right)^{2m}$$

which, for k close to N, approaches to 0 very rapidly when $m \to \infty$. This means that high frequency components get damped very quickly.

This simple analysis clearly justifies the smoothing property that has been observed by numerical experiments.

Other smoothers You may ask if it is just lucky to choose $\sigma = 4/h$ so that the corresponding Richardson method has this smoothing property. There is certain truth to this, since a good choice of damping parameter is not always easy to come by. But it is not so difficult to convince yourself – by the same analysis – that this smoothing property still preserves qualitatively for other choice of σ as long as

$$\sigma \ge \frac{2 + c_0}{h}.$$

where c_0 is constant independent on N.

Note that the above range of σ excludes the case that $\sigma = 2/h$ which corresponds to the so-called ordinary Jacobi method:

$$\text{For } j = 1:N,\ \mu_j^l = \mu_j^{l-1} + \frac{h}{2}\left(\beta_j - \frac{-\mu_{j-1}^{l-1} + 2\mu_j^{l-1} - \mu_{j+1}^{l-1}}{h}\right).$$

This Jacobi method, however, does not have a smoothing property. To see this, with m Jacobi iteration, the Fourier coefficient in front of, for example, the highest frequency is as follows:

$$\begin{aligned} \alpha_N^m &= \left(1 - 2\sin^2 \frac{N\pi}{2(N+1)}\right)^m \alpha_N = \cos^m \frac{N\pi}{N+1}\alpha_N \\ &\sim (-1)^m \left(1 - \frac{\pi^2}{(N+1)^2}\right)^m \alpha_N \end{aligned}$$

which, if $\alpha_N \ne 0$, can not practically get very small for large N.

A parameter-free and yet more efficient smoother is the so-called Gauss-Seidel iteration. This method modifies the Jacobi method in such a way that makes use of the most updated values of each component:

$$\text{For } j = 1:N,\ \mu_j^l = \mu_j^{l-1} + \frac{h}{2}\left(\beta_j - \frac{-\mu_{j-1}^{l} + 2\mu_j^{l-1} - \mu_{j+1}^{l-1}}{h}\right).$$

Comparing it with Jacobi method, the difference lies in the term μ^l_{j-1} in which the superscript l indicates the most updated value of μ_j is used.

It is not hard to imagine that this method has a better convergence property than the Jacobi method. It is also very easy to be convinced by numerical experiments that this is actually a better smoother than the Richardson method, although it is a little bit more complicated to analyze theoretically.

An intuitive discussion Both the Richardson Jacobi and Gauss-Seidel methods are oftentimes called *local relaxation* methods. This name refers to the fact that what both of these algorithms do are just trying to correct the residual vector locally at one nodal point at a time (recall that $\mu_j \approx u(x_j)$). This local relaxation procedure is then effective to the error components that are local in nature. Incidently, the nonsmooth or high frequency component which oscillates across one or few grid points have a strong local feature. Therefore, it is not surprising the both Richardson and Gauss-Seidel iteration can damp out these nonsmooth components more easily. These methods are very inefficient for relatively smoother components in the error since a smoother function is more globally related in nature.

3.2-b Multigrid analysis

In order to fully justify the efficiency of the multigrid method, we will now give a complete proof on the uniform convergence of the multigrid method that we discussed. In general, multigrid convergence analysis is not at all easy. But since we are now dealing with a very special one dimensional problem, it is possible for us to give a rather simple analysis.

The proof we shall present only consists of simple algebraic manipulations. Thus the proof is quite easy to follow logically. But we would like to point out that these algebraic arguments are mostly motivated from analytic considerations. With m throughout this section stays we denote the number of smoothing steps in Algorithm 3.1.18

Error equation Let μ be the exact solution. Then $\beta^k = A_k\mu$ and, from the fine grid smoothing step, we have

$$\mu - \mu^l = (I_k - \sigma_k^{-1}A_k)(\mu - \mu^{l-1}) = \ldots = (I_k - \sigma_k^{-1}A_k)^l\mu.$$

where $I_k \in R^{N_k\times N_k}$ is the identity matrix. By the definition of B_k

$$\begin{aligned}(I_k - B_kA_k)\mu &= \mu - B_k\beta^k = \mu - \mu^m - I^k_{k-1}\epsilon^{k-1}\\ &= (\mu - \mu^m) - I^k_{k-1}B_{k-1}I^{k-1}_kA_k(\mu - \mu^m)\\ &= (I_k - I^k_{k-1}B_{k-1}I^{k-1}_kA_k)(\mu - \mu^m)\\ &= (I_k - I^k_{k-1}B_{k-1}I^{k-1}_kA_k)(I_k - \sigma_k^{-1}A_k)^m\mu.\end{aligned}$$

Since the above relation holds for any μ, we arrive at the following matrix identity:

$$I_k - B_kA_k = (I_k - I^k_{k-1}B_{k-1}I^{k-1}_kA_k)K_k^m,$$

where $K_k = I_k - \sigma_k^{-1} A_k$.

We shall proceed to manipulate the above error equation into a more convenient format. Let us first note that the following basic identity holds:

$$A_{k-1} = I_k^{k-1} A_k I_{k-1}^k.$$

This identity can of course be verified by writing out the matrix multiplication. Another way to check it is to use finite element definition of A_{k-1} and then write the coarse grid finite element basis function in terms of fine grid basis function. Since at least one of these two verifications is straightforward, we leave the details to the readers.(*see the last few lines in the section above*)

Let us write

$$I_k - B_k A_k = (I_k - I_{k-1}^k P_{k-1} + I_{k-1}^k (I_{k-1} - B_{k-1} A_{k-1}) P_{k-1}) K_k^m,$$

where

$$P_{k-1} = A_{k-1}^{-1} I_k^{k-1} A_k.$$

Note that $P_{k-1} : R^{N_k} \mapsto R^{N_{k-1}}$ and it is in some sense a projection matrix with respect to the energy inner project. In fact, the following relation holds for $\mu_k \in R^{N_k}$ and $\nu_{k-1} \in R^{N_{k-1}}$

(3.2.21)
$$(\mu_k, I_{k-1}^k \nu_{k-1})_{A_k} = (P_{k-1}\mu_k, \nu_{k-1})_{A_{k-1}} = (I_{k-1}^k P_{k-1} \mu_k, I_{k-1}^k \nu_{k-1})_{A_k}.$$

Using the above identities, we derive the following identity:

$$\|(I_k - B_k A_k)\mu\|_{A_k}^2 = \|(I_k - I_{k-1}^k P_{k-1})\tilde{\mu}\|_{A_k}^2 + \|(I_{k-1} - B_{k-1} A_{k-1}) P_{k-1} \tilde{\mu}\|_{A_{k-1}}^2$$

where $\tilde{\mu} = K_k^m \mu$.

The following lemma concerns an approximation result which shows how a fine grid vector can be approximated by a coarse grid vector.

Lemma 3.2.22 *For all $\nu \in R^{N_k}$ the following inequality holds:*

$$\|(I_k - I_{k-1}^k P_{k-1})\nu\|_{A_k}^2 \leq \frac{1}{2} h_k \|A_k \nu\|^2.$$

Proof. We first claim that the projection matrix P_{k-1} can be characterized by the following relation

(3.2.23) $$(P_{k-1}\nu)_i = \nu_{2i}.$$

This relation follow from the equality:

$$I_k^{k-1} A_k = A_{k-1} P_{k-1},$$

which can be directly verified.

Let $\mu = (I - I_{k-1}^k P_{k-1})\nu$, we have

$$\mu_{2i} = \nu_{2i} - (P_{k-1}\nu)_i = \nu_{2i} - nu_{2i} = 0,$$

$$\mu_{2i-1} = \nu_{2i-1} - \frac{1}{2}\Big((P_{k-1}\nu)_{i-1} + (P_{k-1}\nu)_i\Big) = \nu_{2i-1} - \frac{1}{2}\Big(\nu_{2i-2} + \nu_{2i}).$$

The proof is then terminated by the next chain of simple relations:

$$\begin{aligned}
\|(I_k - I_{k-1}^k P_{k-1})\nu\|_{A_k}^2 &= \frac{1}{h_k}\sum_{i=1}^{N_k} \mu_i(-\nu_{i-1} + 2\mu_i - \mu_{i+1}) \\
&= \frac{1}{h_k}\sum_{i=1}^{N_k+1} (\mu_i - \mu_{i-1})^2 \\
&= \frac{2}{h_k}\sum_{i=1}^{N_{k-1}} \Big(\nu_{2i-1} - \frac{1}{2}(\nu_{2i-2} + \nu_{2i})\Big)^2 \\
&= \frac{1}{2h_k}\sum_{i=1}^{N_{k-1}} \Big(2\nu_{2i-1} - (\nu_{2i-2} + \nu_{2i})\Big)^2 \\
&\le \frac{1}{2h_k}\sum_{j=1}^{N_k} \Big(2\nu_j - (\nu_{j-1} + \nu_{j+1})\Big)^2 \\
&= \frac{h_k}{2}\|A_k\nu\|^2.
\end{aligned}$$

□

We would like to point out that the relation (3.2.23) is true only for the very special equation (3.1.1). For more general equation, this relation is no longer valid, but the estimate in the lemma still holds (with a different constant in place of $\frac{1}{2}$ in the right hand side of the inequality) and the proof is a little bit more complicate.

The following lemma gives a quantitative estimate on how a fine grid vector smoothed m times by local relaxation can be approximated by a coarse grid vector. It is worth nothing that same norms are used in both hands of the estimate.

Lemma 3.2.24 *Let $\nu \in R^{N_k}$ and $K_k = I_k - \sigma_k^{-1}A_k$. Then the following inequality holds:*

$$\|(I - I_{k-1}^k P_{k-1})K_k^m \nu\|_{A_k}^2 \le \frac{1}{m}(\|\nu\|_{A_k}^2 - \|K_k^m\nu\|_{A_k}^2).$$

Proof. Since $\sigma_k = 4/h_k > \rho(A)$, we have $\sigma(K_k) \subset (0,1)$ and $(K_k\cdot,\cdot)_A = (\cdot, K_k\cdot)_A$. Taking into account these observations and the fact

$$(1-t)t^{2m} \le \frac{1}{2m}(1 - t^{2m}), \quad \forall t \in [0,1],$$

we get:

$$\begin{aligned}
&\|(I - I_{k-1}^k P_{k-1})K_k^m \nu\|_{A_k}^2 \leq \frac{h_k}{2}\|A_k K_k^m \nu\|^2 \\
&= \quad 2(\sigma_k^{-1} A_k K_k^m \nu, A_k K_k^m \nu) = 2((I - K_k)K_k^{2m}\nu, \nu)_{A_k} \\
&\leq \quad \frac{1}{m}\sum_{i=0}^{2m-1}((I - K_k)K_k^i \nu, \nu)_{A_k} \\
&= \quad \frac{1}{m}(((I - K_k^{2m})\nu, \nu)_{A_k}.
\end{aligned}$$

□

Theorem 3.2.25 *Let the operator B_k be defined as in algorithm 3.1.18. Then there exists a real number $\delta \in (0,1)$ depending only on the number of smoothing steps m, such that the following inequality holds*

$$\|(I - B_k A_k)\nu\|_{A_k}^2 \leq \delta \equiv \frac{1}{m+1}\|\nu\|_{A_k}^2 \quad \forall\, \nu \in R^{N_k},$$

for every $k \in \{1, 2, \dots, J\}$.

Proof. The proof is done by induction. For $k = 1$ it trivially follows from the choice $B_1 = A_1^{-1}$. Let us assume that this inequality is true for $k-1$. Then taking $\tilde{\nu}_m = K_k^m \nu$ we have:

$$\begin{aligned}
\|(I - B_k A_k)\nu\|_A^2 &\leq \|(I - P_{k-1})\tilde{\nu}_m\|_{A_k}^2 + \delta\|P_{k-1}\tilde{\nu}_m\|_{A_{k-1}}^2 \\
&= (1 - \frac{1}{m+1})\|(I - P_{k-1})\tilde{\nu}_m\|_{A_k}^2 + \frac{1}{m+1}\|\tilde{\nu}_m\|_{A_k}^2 \\
&\leq \frac{m}{m+1}\frac{1}{m}(\|\nu\|_A^2 - \|\tilde{\nu}_m\|_A^2) + \frac{1}{m+1}\|\tilde{\nu}_m\|_{A_k}^2 \\
&= \frac{1}{m+1}\|\nu\|_{A_k}^2.
\end{aligned}$$

□

(3.2.26) Remark. Looking back at the convergence history plots Figure 3.4 and Figure 3.6 we observe that the numerical experiments show that the estimate obtained in the preceeding theorem is very sharp.

Chapter 4

Conjugate Gradient Methods and Preconditioning

4.1 Condition number

Definition and basic properties

The condition number of a nonsingular operator is usually defined to be $\|A\|\|A^{-1}\|$ with some norm $\|\cdot\|$. For an SPD operator, we shall use the following definition.

Definition 4.1.1 (Condition number) *Given an SPD operator A, its condition number is defined by*

$$\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}.$$

The above definition of the condition number is independent of the choice of norms. We can see easily that such definition results in the smallest possible condition number defined by $\|A\|\|A^{-1}\|$ for any choice of norm $\|\cdot\|$, furthermore, if $\|\cdot\|$ is the norm induced by any inner product $(\cdot,\cdot)$ under which A is self–adjoint, then

$$\kappa(A) = \|A\|\|A^{-1}\|.$$

We say that A is well-conditioned if $\kappa(A)$ is relatively small, otherwise we may say it is ill-conditioned. It is often very difficult to evaluate the condition number exactly. But for most applications, it suffices to get an estimate of its upper bounds. In this respect, the following simple result is useful.

Lemma 4.1.2 *Assume that μ_0 and μ_1 are two positive constants so that*

$$\mu_0(v,v) \le (Av,v) \le \mu_1(v,v) \quad \forall\, v \in \mathcal{V}.$$

Then

$$\kappa(A) \le \frac{\mu_1}{\mu_0}.$$

The condition number is one of the most important parameters associated with a linear algebraic system. The convergence of many iterative methods to be developed depend on the underlying condition number. Roughly speaking, the convergence of a given iterative method is faster if $\kappa(A)$ is smaller. Another example of the role of the condition number can be seen in the following excercise.

(4.1.3) EXERCISE. Assume that

$$Au = f \quad \text{and} \quad A(u + \delta u) = f + \delta f.$$

Prove that

$$\frac{\|\delta u\|}{\|u\|} \le \kappa(A) \frac{\|\delta f\|}{\|f\|}.$$

(4.1.4) EXERCISE. Show that for any two SPD operators $A, B \in L(\mathcal{V})$,

$$\lambda_{\max}(A)\lambda_{\min}(B) \le \lambda_{\max}(AB).$$

$$\kappa(A)/\kappa(B) \le \kappa(AB) \le \kappa(A)\kappa(B).$$

Conditioning of operator product

The following result is useful in the estimate of the condition numbers of the product of two SPD operators.

Lemma 4.1.5 *Assume that A and B are both SPD with respect to $(\cdot,\cdot)$ and μ_0 and μ_1 are two positive constants. The following, that hold for all $v \in \mathcal{V}$, are equivalent:*

1. $\mu_0(Av, v) \le (ABAv, v) \le \mu_1(Av, v)$;
2. $\mu_0(Bv, v) \le (BABv, v) \le \mu_1(Bv, v)$;
3. $\mu_1^{-1}(Av, v) \le (B^{-1}v, v) \le \mu_0^{-1}(Av, v)$;
4. $\mu_1^{-1}(Bv, v) \le (A^{-1}v, v) \le \mu_0^{-1}(Bv, v)$.

If any of the above inequalities holds, then $\kappa(BA) \le \mu_1/\mu_0$.

Proof. Observe that BA, AB, $(BA)^{-1}$ and $(AB)^{-1}$ are self-adjoint with respect to the inner products $(A\cdot,\cdot), (B\cdot,\cdot), (B^{-1}\cdot,\cdot)$ and $(A^{-1}\cdot,\cdot)$, respectively. Hence, by (1.3.7), the above four inequalities are, respectively, equivalent to

1. $\lambda_{\min}(BA) \ge \mu_0$ and $\lambda_{\max}(BA) \le \mu_1$;
2. $\lambda_{\min}(AB) \ge \mu_0$ and $\lambda_{\max}(AB) \le \mu_1$;

3. $\lambda_{\min}((BA)^{-1}) \geq \mu_1^{-1}$ and $\lambda_{\max}((BA)^{-1}) \leq \mu_0^{-1}$;
4. $\lambda_{\min}((AB)^{-1}) \geq \mu_1^{-1}$ and $\lambda_{\max}((AB)^{-1}) \leq \mu_0^{-1}$;

The desired results then follow from the obvious relations between the spectrum of

$$BA, AB, (BA)^{-1}, \quad \text{and} \quad (BA)^{-1}.$$

□

Lemma 4.1.6 *Assume that A and B are both SPD with respect to $(\cdot,\cdot)$ and $\delta \in (0,1)$. The following, that hold for all $v \in \mathcal{V}$, are equivalent:*

1. $0 \leq ((I - BA)v, v)_A \leq \delta(v,v)_A$.

2. $0 \leq ((I - BA)v, v)_{B^{-1}} \leq \delta(v,v)_{B^{-1}}$.

3. $0 \leq ((B^{-1} - A)v, v) \leq \frac{\delta}{1-\delta}(v,v)_A$.

4.2 The conjugate gradient method

4.2-a Gradient method and steepest descent method

The starting point of this sort of method is the fact that u is the solution of $Au = f$ if any only it is the minimizer of the following quadratic functional:

$$J(v) = \frac{1}{2}(Av, v) - (f, v).$$

Namely,

$$J(u) = \min_{v \in V} J(v).$$

The above fact can be easily seen from the following simple relation

$$J(v) = J(u) + (Au - f, v) + \frac{1}{2}(Av, v), \quad u, v \in V.$$

This identity can also be viewed as a Taylor expasion since

$$(\nabla J)(u) = Au - f, (\nabla^2 J)(u) = A.$$

An iterative method for solving $Au = f$ can then be obtained by solving the minimization problem iteratively. One simple and natural strategy is the gradient method, which is based on a general fact that a function at a given point decrease most rapidely in the direction of negative gradient. Assume that u_k is the $k-$th iterate. At u_k, the function J decrease most rapidly in the direction of negative gradient $-\nabla J(u_k) = r_k$ where $r_k = f - Au_k$ is the residual. Thus, if $r_k \neq 0$, we can obtain $u_{k+1} = u_k + \alpha_k r_k$ with α_k determined by

$$J(u_k + \alpha_k r_k) = \min_{\alpha \in \mathbb{R}^1} J(u_k + \alpha r_k).$$

It is easy to see that

$$\alpha_k = (r_k, r_k)/(Ar_k, r_k).$$

Algorithm 4.2.7 (Gradient Method)

```
*Given* $u_0$*;* $r_0 = f - Au_0$*;*
**for** $k = 0, 1, \ldots$ ,till convergence
  $\alpha_k = (r_k, r_k)/(Ar_k, r_k)$;
  $u_{k+1} = u_k + \alpha_k r_k$;
  $r_{k+1} = r_k - \alpha_k A r_k$;
**endfor**
```

It is easy to see the above steepest descent method satisfies the following best approximation property:

$$\|u_{k+1} - u\|_A = \min_{\alpha \in \mathbb{R}^1} \|(u_k + \alpha r_k) - u\|_A \tag{4.2.8}$$

which follows directly from the simple identity:

$$\|v - u\|_A^2 = 2J(v) + (f, u), \quad \forall v \in V.$$

Consequently

$$\|u_{k+1} - u\|_A = \min_{\alpha \in \mathbb{R}^1} \|(I - \alpha A)(u_k - u)\|_A \le \min_{\alpha \in \mathbb{R}^1} \|I - \alpha A\|_A \, \|(u_k - u)\|_A.$$

Notice that

$$\min_{\alpha \in \mathbb{R}^1} \|I - \alpha A\|_A = \min_{\alpha \in \mathbb{R}^1} \max_{\lambda \in \sigma(A)} |1 - \alpha \lambda| = \frac{\lambda_{\max}(A) - \lambda_{\min}(A)}{\lambda_{\max}(A) + \lambda_{\min}(A)}.$$

Therefore we have obtained the following convergence estimate for the steepest descent method.

$$\|u - u_k\|_A \le \left(\frac{\kappa(A) - 1}{\kappa(A) + 1}\right)^k \|u - u_0\|_A. \tag{4.2.9}$$

(4.2.10) EXERCISE. Prove that for the steepest descent method

$$(r_{k+1}, r_k) = 0 \quad k = 0, 1, 2, \ldots.$$

The steepest descent method can be regarded as an iterative procedure that minimizes J along the search directions $\{r_0, r_1, r_2, \ldots ...\}$ successively. Let us look at this kind of method in a more general perspective. Namely we consider a general iterative procedure that minimizes J along some given search directions $\{p_0, p_1, p_2, \ldots\}$ successively. The $k + 1$st iterate is given by

$$u_{k+1} = u_k + \alpha_k p_k.$$

A very desirable situation is that not only u_{k+1} is optimal along the direction p_k:

$$J(u_{k+1}) = \min_{\alpha \in \mathbb{R}^1} J(u_k + \alpha p_k)$$

but also optimal along all the previous search directions:

$$J(u_{k+1}) = \min_{v \in V_k} J(v)$$

where

$$V_k \equiv \mathrm{span}(p_0, p_1, p_2, \ldots, p_k).$$

It is easy to see that the aforementioned desired property can indeed be valid if the search directions are mutually orthogonal with respect to $A-$inner product:

$$(Ap_i, p_j) = 0 \quad i \neq j.$$

4.2-b Derivation of the basic algorithm

The conjugate gradient method is closely related to the the Krylov spaces defined as follows:

$$\mathcal{V}_0 = 0 \quad \text{and} \quad \mathcal{V}_k = \mathrm{span}\{f, Af, \cdots, A^{k-1}f\}, \quad 1 \leq k \leq n.$$

We introduce the A-orthogonal compliment of $\mathcal{V}_k \subset \mathcal{V}_{k+1}$

$$W_k = \{w \in \mathcal{V}_{k+1}, (Aw, v_k) = 0, \forall v_k \in \mathcal{V}_k\}.$$

It is straightforward to see that if $\mathcal{V}_{k-1} = \mathcal{V}_k$ then $\mathcal{V}_k = \mathcal{V}_{k+1}$. Hence $W_{k-1} = 0$ implies $W_k = 0$. Therefore there exists an integer $m \leq n$ such that

$$\mathcal{V}_n = \mathcal{V}_m = W_0 \oplus W_2 \oplus \cdots \oplus W_{m-1}$$

with

$$W_i = \mathrm{span}\{p_i\} \neq 0, \quad 1 \leq i \leq m-1.$$

Here $p_i \in W_i \setminus \{0\}$. By definition of W_i

(4.2.11) $$(Ap_i, p_j) = 0, \quad \text{if} \quad i \neq j.$$

Notice that, for $0 \leq i \leq m-1$

(4.2.12) $$\mathcal{V}_i = W_0 \oplus W_1 \oplus \cdots \oplus W_{i-1}.$$

Observe that the solution of $Au = f$ satisfies $u \in \mathcal{V}_n = \mathcal{V}_m$, hence

$$u = \sum_{i=0}^{m-1} \alpha_i p_i$$

for some constants α_i which, by orthogonality, can be determined by

(4.2.13) $$\alpha_i = \frac{(Au, p_i)}{(Ap_i, p_i)} = \frac{(f, p_i)}{(Ap_i, p_i)}.$$

The problem is that the vetors p_i are not known in advance, otherwise the exact solution would have been obtained directly this way. We shall now demonstrate how to compute these vectors in a recursive fashion. Obviously, we can take $p_0 = f$. Assuming p_i are known for $0 \le i \le k$. We need to figure out how to compute p_{k+1}.

We denote

$$u_i = \sum_{j=1}^{i-1} \alpha_j p_j, r_i = f - Au_i.$$

Obviously $u_i \in \mathcal{V}_i, r_i \in \mathcal{V}_{i+1}$ and

$$(Au_i, v) = (Au, v) = (f, v), \quad \text{or} \quad (r_i, v) = 0 \quad \forall\, v \in \mathcal{V}_i.$$

It follows that

$$(r_i, r_j) = 0, \quad i \neq j$$

and

(4.2.14) $$(r_{k+1}, p_i) = 0, \quad 0 \le i \le k, \quad \text{or} \quad r_{k+1} \perp \mathcal{V}_{k+1}$$

which implies (since $A\mathcal{V}_k \subset \mathcal{V}_{k+1}$)

(4.2.15) $$(Ar_{k+1}, p_i) = (r_{k+1}, Ap_i) = 0, \quad 0 \le i \le k-1, \quad \text{or} \quad r_{k+1} \perp_A \mathcal{V}_k$$

Armed with the above identities, we are now ready to compute p_{k+1}. If $r_{k+1} = 0$, then there is no need for us to proceed since $u_{k+1} = u$ is the exact solution. Now we assume $r_{k+1} \neq 0$. In this case, $r_{k+1} \not\in \mathcal{V}_{k+1}$ (see (4.2.14)), $r_{k+1} \in \mathcal{V}_{k+2}$. This means that $\mathcal{V}_{k+2} = \operatorname{span}(r_{k+1}, \mathcal{V}_{k+1})$, or $\{p_0, p_1, \ldots, p_k, r_{k+1}\}$ form a basis for $\mathcal{V}_{k+2}$. Therefore $p_{k+1} \in \mathcal{V}_{k+2}$ can be represented as

$$p_{k+1} = r_{k+1} + \sum_{i=0}^{k} \tilde{\beta}_i p_i.$$

By definition, p_{k+1} must satisfy $(Ap_{k+1}, p_i) = 0$ for $0 \le i \le k$. It thus follows from (4.2.15) that

$$\tilde{\beta}_i = 0 \quad (0 \le i \le k-1) \quad \text{and} \quad \tilde{\beta}_k = -\frac{(Ar_{k+1}, p_k)}{(Ap_k, p_k)}.$$

With a slightly different notation for $\tilde{\beta}_k$, we have

$$p_{k+1} = r_{k+1} + \beta_k p_k, \quad \text{with } \beta_k = -\frac{(Ar_{k+1}, p_k)}{(Ap_k, p_k)}.$$

Consequently, if we set $u_0 = 0$ and $p_0 = 0$, then for $k = 1, 2, \ldots,$

$$u_{k+1} = u_k + \alpha_k p_k$$

(4.2.16) $$r_{k+1} = r_k - \alpha_k A p_k$$

(4.2.17) $$p_{k+1} = r_{k+1} + \beta_k p_k.$$

These identities represent a verion of the well-known conjugate gradient methods.

We shall derive some alternative (more efficient) formulae for α_k and β_k.

It can be easily proved that

$$\alpha_k = \frac{(r_k, r_k)}{(Ap_k, p_k)}.$$

$$\beta_k = \frac{(r_{k+1}, r_{k+1})}{(r_k, r_k)}.$$

We are now in a position to state the conjugate gradient method. We assume that an initial guess u_0 is given and can then apply the agorithm derived above to the residue equation $Av = f - Au_0$. The resultant algorithm is as follows.

Algorithm 4.2.18 (Conjugate Gradient Method)

```
Given $u_0$; $r_0 = f - Au_0$; $p_0 = r_0$;
for $k = 0, 1, \ldots$ ,till convergence
    $\alpha_k = (r_k, r_k)/(Ap_k, p_k)$;
    $u_{k+1} = u_k + \alpha_k p_k$;
    $r_{k+1} = r_k - \alpha_k Ap_k$;
    $\beta_k = (r_{k+1}, r_{k+1})/(r_k, r_k)$;
    $p_{k+1} = r_{k+1} + \beta_k p_k$.
endfor
```

Notice that only one operator and vector multiplication and two vector inner products are needed in above algorithm in each iteration.

Convergence estimate

We know that $u_m = u$, hence the conjugate gradient method gives rise to the exact solution after m-th iteration. This means that CG method can be regarded as a direct method. The most important feature of the method, however, is that it can be regarded as an iterative method as well. We shall now show that the error is reducing at each iteration.

The following simple result follows from (4.2.14) and (4.2.15).

Lemma 4.2.19 *Assume that u_k is the k-th iteration of CG method*

$$(Au_k, v) = (f, v) \quad \forall\, v \in \mathcal{V}_k.$$

$$(A(u - u_k), u - u_k) \le (A(u - v), u - v) \quad \forall\, v \in V_k.$$

(4.2.20) EXERCISE. Prove Lemma (4.2.19).

(4.2.21) EXERCISE. Show that $Au = f$ iff $J(u) = \inf_{v \in \mathcal{V}} J(v)$ where $J(v) = \frac{1}{2}(Av, v) - (f, v)$.

(4.2.22) EXERCISE. Assume u_k is the k-th iteration of CG method, show that

$$J(u_k) = \inf_{v \in \mathcal{V}_k} J(v).$$

Theorem 4.2.23

$$\|u-u_k\|_A \le 2\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^k \|u-u_0\|_A$$

Proof. For an arbitrary polynomial p_{k-1} of degree $k-1$, taking $v = p_{k-1}(A)f = Ap_{k-1}(A)u$, we derive from (4.2.19) that

$$\begin{aligned}
(A(u-u_k), u-u_k) &\le \min_{p_{k-1}}(A(I-Ap_{k-1}(A))f, (I-Ap_{k-1}(A))u) \\
&\le \min_{q_k(0)=1}(Aq_k(A)u, q_k(A)u) \\
&\le \min_{q_k(0)=1}\max_{\lambda\in\sigma(A)}|q_k(\lambda)|^2(Au,u) \\
&\le \min_{q_k(0)=1}\max_{\lambda\in[a,b]}|q_k(\lambda)|^2(Au,u).
\end{aligned}$$

Here $a = \lambda_{\min}(A)$ and $b = \lambda_{\max}(A)$.

We choose

$$\tilde{q}_k(\lambda) = \frac{T_k(\frac{b+a-2\lambda}{b-a})}{T_k(\frac{b+a}{b-a})}.$$

Here $T_k(t)$ is the Tschebyshev polynomial of degree k given by

$$T_k(t) = \begin{cases} \cos(k\cos^{-1}t)) & \text{if} \quad |t|\le 1; \\ (\text{sign}(t))^k\cosh(k\cosh^{-1}t)) & \text{if} \quad |t|\ge 1; \end{cases} \tag{4.2.24}$$

Notice that $T_k(\frac{b+a-2\lambda}{b-a}) \le 1$ for $\lambda\in[a,b]$. Thus

$$\max_{\lambda\in[a,b]}|\tilde{q}_k(\lambda)| \le \left[T_k(\frac{b+a}{b-a})\right]^{-1}.$$

We set

$$\frac{b+a}{b-a} = \cosh\sigma = \frac{e^\sigma + e^{-\sigma}}{2}.$$

Solving this equation for e^σ, we have

$$e^\sigma = \frac{\sqrt{\kappa(A)}+1}{\sqrt{\kappa(A)}-1}$$

since $\kappa(A) = b/a$. We then obtain

$$\cosh k\sigma = \frac{e^{k\sigma}+e^{-k\sigma}}{2} \ge \frac{1}{2}e^{k\sigma} = \frac{1}{2}\left(\frac{\sqrt{\kappa(A)}+1}{\sqrt{\kappa(A)}-1}\right)^k.$$

Consequently

$$\min_{q_k(0)=1}\max_{\lambda\in[a,b]}|q_k(\lambda)| \le 2\left(\frac{\sqrt{\kappa(A)}-1}{\sqrt{\kappa(A)}+1}\right)^k.$$

The desired result then follows. $\square$

Even though, the estimate given in above theorem is sufficient for many applications but in general it is not sharp. There are many ways to sharpen the estimate. For example, the following improved estimate shows that the convergence of the CG method depends on the distribution of the spectrum of A. It is possible that the CG method converges fast even the condition number of A is large.

Theorem 4.2.25 *Assume that $\sigma(A) = \sigma_0(A) \cup \sigma_1(A)$ and l is the number of elements in $\sigma_0(A)$. Then*

$$\|u - u_k\|_A \le 2M \left(\frac{\sqrt{b/a} - 1}{\sqrt{b/a} + 1} \right)^{k-l} \|u - u_0\|_A$$

where $a = \min_{\lambda \in \sigma_1(A)} \lambda$, $b = \max_{\lambda \in \sigma_1(A)} \lambda$ and

$$M = \max_{\lambda \in \sigma_1(A)} \prod_{\mu \in \sigma_0(A)} \left| 1 - \frac{\lambda}{\mu} \right| .$$

Proof. The proof is almost identical to that for the previous theorem except that we take

$$\tilde{q}_k(\lambda) = \frac{T_{k-l}(\frac{b+a-2\lambda}{b-a})}{T_{k-l}(\frac{b+a}{b-a})} \prod_{\mu \in \sigma_0(A)} \left(1 - \frac{\lambda}{\mu} \right) .$$

$\square$

(4.2.26) EXERCISE. Verify that the functions $T_k(t)$ given by (4.2.24) are indeed polynomials of degree k. *hint:* Use the formula

$$\cos(k+1)x = -\cos(k-1)x + 2\cos kx \cos x$$

and similar formula for $\cosh(k+1)x$ to establish a recurrence relation between $T_k(t)$.

4.3 Preconditioner and PCG

Basic ideas and algorithm

We notice that the above estimate for the convergence rate of the CG method strongly depends on the condition number. If $\kappa(A)$ is too large, the convergence can be very slow. PCG method is just designed to reduce the magnitude of the original condition number and hence improve the speed of the convergence. More specifically, we find another SPD operator B on $\mathcal{V}$ (which is self-adjoint with respect to the same inner product as for A), and consider the preconditioned equation:

$$BAu = Bf. \tag{4.3.27}$$

Introducing a new inner product $[u,v] = (B^{-1}u, v)$ then BA is SPD under this new inner product since $[BAu, v] = (Au, v)$. Therefore the CG method may be applied to (4.3.27) under the new inner product and resultant algorithm is called the *preconditioned conjugate gradient method.*

Algorithm 4.3.28 (Preconditioned conjugate gradient)

```
Given $u_0$; $r_0 = f - Au_0$; $p_0 = Br_0$;
for $k = 1, \dots$ ,till convergence
  $\alpha_k = (Br_{k-1}, r_{k-1})/(Ap_{k-1}, p_{k-1})$;
  $u_k = u_{k-1} + \alpha_k p_{k-1}$;
  $r_k = r_{k-1} - \alpha_k A p_{k-1}$;
  $\beta_k = (Br_k, r_k)/(Br_{k-1}, r_{k-1})$;
  $p_k = Br_k + \beta_k p_{k-1}$.
endfor
```

Stopping criteria shold be discussed here.

Note that, in each iteration, only one evaluation of the action of A and B and two vector inner products are needed in this algorithm. If $B = I$, this coincides with the usual CG method.

The operator B in the above is often known as a *preconditioner.* The extreme choices of B are $B = A^{-1}$ and $B = I$. In the former case, we get the smallest possible condition number 1, but of course the action of A^{-1} is most difficult to compute. For $B = I$, the action of B is trivial but nothing happens. Roughly speaking, a good precontioner should be someway in bebween A^{-1} and I. Namely, the action of B is easy to compute and also BA should be better conditioned, i.e. $\kappa(BA)$ is smaller than $\kappa(A)$. How "easy" or how "small" is relative and depends on the background of the underlying problem. We would like to warn that, in our definition of preconditioner, B is the operator that spectrally behaves like A^{-1} instead of A itself.

Subspace preconditioning theorems

Two thereoms will be given here.

Preconditioner and linear iterations

Proposition 4.3.29 *Assume that B is symmetric with respect to the inner product $(\cdot,\cdot)$. If the scheme* (2.1.1) *is convergent, then B is SPD and*

$$\kappa(BA) \le \frac{1+\rho}{1-\rho},$$

where $\rho = \rho(I - BA) < 1$.

(4.3.30) EXERCISE. Prove Proposition (4.3.29).

PCG accelaration

Notice that the convergence rate of scheme (2.1.1) is ρ, but if we use B from the scheme as a preconditioner for A, the PCG method converges at a faster rate

$$\delta = \frac{\sqrt{\kappa(BA)} - 1}{\sqrt{\kappa(BA)} + 1} \le \frac{\sqrt{\frac{1+\rho}{1-\rho}} - 1}{\sqrt{\frac{1+\rho}{1-\rho}} + 1} = \frac{1 - \sqrt{1-\rho^2}}{\rho} < \rho.$$

We conclude that for any linear iterative scheme (2.1.1) for which B is symmetric, a preconditioner for A can be attained and the convergence rate of (2.1.1) can be accelerated by using the PCG method. For example, a preconditioner can be resulted from SSOR method as follows

$$B = SS^t, \quad S = (D - \omega U)D^{\frac{1}{2}}.$$

A more interesting case is that the scheme (2.1.1) may not be convergent at all whereas B can always be a preconditioner. For example, the Jacobi method is not convergent for all SPD system, but $B = D^{-1}$ can always be used as a preconditioner. This preconditioner is often known as diagonal preconditioner.

On the other hand any preconditioner of A can be used to construct a linear iterative scheme, as indicated in the following

Proposition 4.3.31 *Assume that B is a preconditioner of A. Then the following linear iteration*

$$u^{k+1} = (I - \omega BA)u_{k-1} + \omega Bf$$

is convergent for $\omega \in (0,\ 2/\rho(BA))$ and the optimal convergence rate is attained when $\omega = 2(\lambda_{\min}(BA) + \lambda_{\max}(BA))^{-1}$ and results in a reduction of $(\kappa(BA) - 1)/(\kappa(BA) + 1)$ per iteration.

The proof of the above proposition is left to the interested reader.

(4.3.32) EXERCISE. Prove (4.3.31).

Chapter 5

Iterative methods by subspace correction

Following Xu (1992), a general framework of constructing linear iterative methods and/or preconditioners can be obtained by the concept of *space decomposition* and *subspace correction*. As we shall see late, most major iterative methods including many basic multigrid and domain decomposition methods fall into this framework. This framework will be presented here from a purely algebraic point of view. Some simple examples are given for illustration and more important applications are given in the later chapters for multigrid and domain decomposition methods.

5.1 Preliminaries

A decomposition of a vector space $\mathcal{V}$ consists of a number of subspaces $\mathcal{V}_i \subset \mathcal{V}$ (for $0 \le i \le J$) such that

(5.1.1)
$$\mathcal{V} = \sum_{i=0}^{J} \mathcal{V}_i.$$

This means that, for each $v \in \mathcal{V}$, there exist $v_i \in \mathcal{V}_i$ ($0 \le i \le J$) such that $v = \sum_{i=0}^{J} v_i$. This representation of v may not be unique in general, namely (5.1.1) is not necessarily a direct sum.

For each i, we define $Q_i, P_i : \mathcal{V} \mapsto \mathcal{V}_i$ and $A_i : \mathcal{V}_i \mapsto \mathcal{V}_i$ by

(5.1.2)
$$(Q_i u, v_i) = (u, v_i), \quad (P_i u, v_i)_A = (u, v_i)_A, \quad u \in \mathcal{V}, v_i \in \mathcal{V}_i,$$

and

(5.1.3)
$$(A_i u_i, v_i) = (A u_i, v_i), \quad u_i, v_i \in \mathcal{V}_i.$$

Q_i and P_i are both orthogonal projections and A_i is the restriction of A on $\mathcal{V}_i$ and is SPD. It follows from the definition that

$$A_i P_i = Q_i A. \tag{5.1.4}$$

This identity is of fundamental importance and will be used frequently in this chapter. A consequence of it is that, if u is the solution of (2.0.1), then

$$A_i u_i = f_i \tag{5.1.5}$$

with $u_i = P_i u$ and $f_i = Q_i f$. This equation may be regarded as the restriction of (2.0.1) to $\mathcal{V}_i$.

We note that the solution u_i of (5.1.5) is the best approximation of the solution u (2.0.1) in the subspace $\mathcal{V}_i$ in the sense that

$$J(u_i) = \min_{v \in \mathcal{V}_i} J(v), \quad \text{with } J(v) = \frac{1}{2}(Av, v) - (f, v)$$

and

$$\|u - u_i\|_A = \min_{v \in \mathcal{V}_i} \|u - v\|_A.$$

The subspace equation (5.1.5) will be in general solved approximately. To describe this, we introduce, for each i, another non-singular operator $R_i : \mathcal{V}_i \mapsto \mathcal{V}_i$ that represents an approximate inverse of A_i in certain sense. Thus an approximate solution of (5.1.5) may be given by $\hat{u}_i = R_i f_i$.

(5.1.6) Example. Consider the space $\mathcal{V} = R^n$ and the simplest decomposition:

$$\mathbb{R}^n = \sum_{i=1}^{n} \operatorname{span}\{e^i\},$$

where e^i is the i-th column of the identity matrix. For a SPD matrix $A = (a_{ij}) \in \mathbb{R}^{n\times n}$

$$A_i = a_{ii}, \quad Q_i y = y_i e^i,$$

where y_i the i-th component of $y \in \mathbb{R}^n$.

5.2 Basic algorithms

From the viewpoint of subspace correction, most linear iterative methods can be classified into two major algorithms, namely the *parallel subspace correction* (PSC) method and the *successive subspace correction* method (SSC).

PSC: Parallel subspace correction This type of algorithm is similar to Jacobi method. The idea is to correct the residue equation on each subspace in parallel.

Let u^{old} be a given approximation of the solution u of (2.0.1). The accuracy of this approximation can be measured by the residual: $r^{\text{old}} = f - Au^{\text{old}}$.

If $r^{\text{old}} = 0$ or very small, we are done. Otherwise, we consider the residual equation:

$$Ae = r^{\text{old}}.$$

Obviously $u = u^{\text{old}} + e$ is the solution of (2.0.1). Instead we solve the restricted equation to each subspace $\mathcal{V}_i$

$$A_i e_i = Q_i r^{\text{old}}.$$

It should be helpful to note that the solution e_i is the best possible correction u^{old} in the subspace $\mathcal{V}_i$ in the sense that

$$J(u^{\text{old}} + e_i) = \min_{e \in \mathcal{V}_i} J(u^{\text{old}} + e), \quad \text{with } J(v) = \frac{1}{2}(Av, v) - (f, v)$$

and

$$\|u - (u^{\text{old}} + e_i)\|_A = \min_{e \in \mathcal{V}_i} \|u - (u^{\text{old}} + e)\|_A.$$

As we are only seeking for a correction, we only need to solve this equation approximately using the subspace solver R_i described earlier

$$\hat{e}_i = R_i Q_i r^{\text{old}}.$$

An update of the approximation of u is obtained by

$$u^{new} = u^{\text{old}} + \sum_{i=0}^{J} \hat{e}_i$$

which can be written as

$$u^{new} = u^{\text{old}} + B(f - Au^{\text{old}}),$$

where

$$B = \sum_{i=0}^{J} R_i Q_i. \tag{5.2.7}$$

We have therefore

Algorithm 5.2.8 *Given $u_0 \in \mathcal{V}$, apply the iterative secheme* (2.1.1) *with B given in* (5.2.7).

(5.2.9) Example. With $\mathcal{V} = \mathbb{R}^n$ and the decomposition given by 5.1, the corresponding (5.2.8) is just the Jacobi iterative method.

It is well-known that the Jacobi method is not convergent for all SPD problems, hence (5.2.8) is not always convergent. However the preconditioner obtained from this algorithm is of great importance.

Lemma 5.2.10 *The operator B given by* (5.2.7) *is SPD if each $R_i : \mathcal{V}_i \to \mathcal{V}_i$ is SPD.*

Proof. The symmetry of B follows from the symmetry of R_i. Now, for any $v \in \mathcal{V}$, we have $(Bv, v) = \sum_{i=0}^{J}(R_i Q_i v, Q_i v) \geq 0$. If $(Bv, v) = 0$, we then have $Q_i v = 0$ for all i. Let $v_i \in \mathcal{V}_i$ be such that $v = \sum_i v_i$, then $(v, v) = \sum_i (v, v_i) = \sum_i (Q_i v, v_i) = 0$. Therefore $v = 0$ and B hence is positive and definite. □

Algorithm 5.2.11 *Apply the CG method to equation* (2.0.1)*, with B defined by* (5.2.7) *as a preconditioner.*

(5.2.12) Example. The preconditioner B corresponding to 5.1 is

$$B = \mathrm{diag}(a_{11}^{-1}, a_{22}^{-1}, \cdots, a_{nn}^{-1})$$

which is the well-known diagonal preconditioner for the SPD matrix A.

SSC: Successive subspace correction This type of algorithm is similar to the Gauss-Seidel method.

To improve the PSC method that makes simultaneous correction, we here make the correction in one subspace at a time by using the most updated approximation of u. More precisely, starting from $v^{-1} = u^{\mathrm{old}}$ and correcting its residule in $\mathcal{V}_0$ gives

$$v^0 = v^{-1} + R_0 Q_0 (f - Av^{-1}).$$

By correcting the new approximation v^1 in the next space $\mathcal{V}_1$, we get

$$v^1 = v^0 + R_1 Q_1 (f - Av^0).$$

Proceeding this way successively for all $\mathcal{V}_i$ leads to

Algorithm 5.2.13 *Given $u^0 \in \mathcal{V}$.*

```
for $k = 0, 1, \ldots$ till convergence
  $v \leftarrow u^k$
  for $i = 0 : J$   $v \leftarrow v + R_i Q_i (f - Av)$   endfor
  $u^{k+1} \leftarrow v$.
endfor
```

(5.2.14) Example. Corresponding to decomposition in Example 5.1, the Algorithm 5.2.13 is the Gauss-Seidel iteration.

(5.2.15) Example. Gaussian elimination is an "exact" SSC method with projection defined with respect to $(\cdot,\cdot)_{A^t A}$. `details checked?`.

(5.2.16) Example. More generally, decompose $\mathbb{R}^n$ as

$$\mathbb{R}^n = \sum_{i=0}^{J} \mathrm{span}\{e^{l_i}, e^{l_i+1}, \cdots, e^{l_{i+1}-1}\},$$

where $1 = l_0 < l_1 < \cdots < l_{J+1} = n+1$. Then (5.2.8), (5.2.11) and (5.2.13) are the block Jacobi method, block diagonal preconditioner and block Gauss-Seidel method respectively.

Let $T_i = R_i Q_i A$. By (5.1.4), $T_i = R_i A_i P_i$. Note that $T_i : \mathcal{V} \mapsto \mathcal{V}_i$ is symmetric with respect to $(\cdot,\cdot)_A$ and nonnegative and that $T_i = P_i$ if $R_i = A_i^{-1}$.

If u is the exact solution of (2.0.1), then $f = Au$. Let v^i be the $i-th$ iterate (with $v^0 = u^k$) from Algorithm 5.2.13, we have by definition

$$u - v^{i+1} = (I - T_i)(u - v^i), \quad i = 0, \cdots, J.$$

A successive application of this identity yields

$$u - u^{k+1} = E_J(u - u^k), \tag{5.2.17}$$

where

$$E_J = (I - T_J)(I - T_{J-1}) \cdots (I - T_1)(I - T_0). \tag{5.2.18}$$

(5.2.19) Remark. It is interesting to look at the operator E_J in the special case that $R_i = \omega A_i^{-1}$ for all i. The corresponding SSC iteration is a generalization of the classic SOR method. In this case, we have

$$E_J = (I - \omega P_J)(I - \omega P_{J-1}) \cdots (I - \omega P_1)(I - \omega P_0).$$

One trivial fact is that E_J is invertible when $\omega \neq 1$. Following an argument by Nicolaides (1973) for the SOR method, let us take a look at the case $\omega = 2$. Since, obviously, $(I - 2P_i)^{-1} = (I - 2P_i)$ for each i, we conclude that $E_J^{-1} = E_J^*$ where $*$ is the adjoint with respect to the inner product $(\cdot,\cdot)_A$. This means that E_J is an orthogonal operator and, in particular, $\|E_J\|_A = 1$. As a consequence, the SSC iteration can not converge when $\omega = 2$. In fact, as we shall see in Proposition **??**, in this special case, that the SSC method converges if and only if $0 < \omega < 2$.

Algorithm (5.2.13) can also be symmetrized.

```
Algorithm 5.2.20  Given $u^0 \in \mathcal{V}$, $v \leftarrow u^0$
  for $k = 0, 1, \ldots$ till convergence
    for $i = 0 : J$ and $i = J : -1 : 0$    $v \leftarrow v + R_i Q_i (f - Av)$   endfor
  endfor
```

The advantage of the symmetrized algorithm is that it can be used as a preconditioner. In fact, (5.2.20) can be formulated in the form of (2.1.1) with operator B defined as follows: For $f \in \mathcal{V}$, let $Bf = u^1$ with u^1 obtained by (5.2.20) applied to (2.0.1) with $u^0 = 0$.

Similar to Young's SOR method, let us introduce a relaxation method.

```
Algorithm 5.2.21  Given $u^0 \in \mathcal{V}$, $v \leftarrow u^0$
  for $k = 0, 1, \ldots$ till convergence
    for $i = 0 : J$   $v \leftarrow v + \omega R_i Q_i (f - Av)$   endfor
  endfor
```

With $\mathcal{V} = \mathbb{R}^n$ and the decomposition given in Example 5.1, the above algorithm is the SOR method. Like the SOR method, that a proper choice of ω can result in an improvement of the convergence rate, but it is not easy to find an optimal ω in general. The above algorithm is essentially the same as (5.2.13) since we can absorb the relaxation parameter ω into the definition of R_i.

Colorization and parallelization of SSC iteration Associated with a given partition (5.1.1), a coloring of the set $\mathcal{J} = \{0, 1, 2, \ldots, J\}$ is a disjoint decomposition:

$$\mathcal{J} = \uplus_{t=1}^{J_c} \mathcal{J}(t)$$

such that

$$P_i P_j = 0 \quad \text{for any} \quad i, j \in \mathcal{N}(t), i \neq j (1 \leq t \leq J_c).$$

We say that i, j have the same color if they both belong to some $\mathcal{J}(t)$.

The important property of the coloring is that the SSC iteration can be carried out in parallel in each color.

Algorithm 5.2.22 (Colored SSC) *Given* $u^0 \in \mathcal{V}$, $v \leftarrow u^0$

```
for $k = 0, 1, \ldots$ till convergence
  for $t = 1 : J_c$   $v \leftarrow v + \sum_{i \in \mathcal{J}(t)} R_i Q_i (f - Av)$   endfor
endfor
```

We note that the terms under the sum in the above algorithm can be evaluated in parallel (for each t, namely within the same color).

5.3 Basic convergence results

The purpose of this section is to establish an abstract theory for algorithms described in previous sections. For reasons mentioned in Remarks 3.2 and 3.4, it suffices to study Algorithms 5.2.11 and 5.2.13. Two fundamental theorems will be presented.

For Algorithm 5.2.11, we need to estimate the condition number of

$$T = BA = \sum_{i=1}^{J} T_i,$$

where B is defined by (5.2.7) and $T_i = R_i A_i P_i$.

For Algorithm 5.2.13, we need to establish the contraction property: there exists a constant $0 < \delta < 1$ such that

$$\|E_J\|_A \leq \delta \quad \text{with} \quad \|E_J\|_A = \sup_{v \in \mathcal{V}} \frac{\|E_J v\|_A}{\|v\|_A},$$

where E_J is given by (5.2.18), or equivalently

$$\|v\|_A^2 \leq \frac{1}{1 - \delta^2} (\|v\|_A^2 - \|E_J v\|_A^2), \quad \forall\, v \in \mathcal{V}. \tag{5.3.1}$$

Applying this estimate to (5.2.17) yields $\|u-u^k\|_A \le \delta^k \|u-u^0\|_A$.

The estimates of $\kappa(BA)$ and $\|E_J\|_A$ are mainly in terms of two parameters, K_0 and K_1, defined as follows.

1. For any $v \in \mathcal{V}$, there exists a decomposition $v = \sum_{i=1}^J v_i$ for $v_i \in \mathcal{V}_i$ such that

$$\sum_{i=1}^J (R_i^{-1} v_i, v_i) \le K_0 (Av, v). \tag{5.3.2}$$

2. For any $S \subset \{1, 2, \cdots J\} \times \{1, 2, \cdots J\}$ and $u_i, v_i \in \mathcal{V}$ for $i = 1, 2, \cdots, J$,

$$\sum_{(i,j)\in S} (T_i u_i, T_j v_j)_A \le K_1 \left(\sum_{i=1}^J (T_i u_i, u_i)_A\right)^{\frac{1}{2}} \left(\sum_{j=1}^J (T_j v_j, v_j)_A\right)^{\frac{1}{2}}. \tag{5.3.3}$$

5.3-a Fundamental theorems

Our first fundamental theorem is an estimate of the condition number of BA.

Theorem 5.3.4 (Fundamental theorem I) *Assume that B is the SSC preconditioner given by* (5.2.7)*; then*

$$\kappa(BA) \le K_0 K_1.$$

Proof. We prove that

$$\lambda_{\max}(BA) \le K_1, \tag{5.3.5}$$

and

$$\lambda_{\min}(BA) \ge K_0^{-1}. \tag{5.3.6}$$

It follows directly from the definition of K_1 that

$$\|Tv\|_A^2 = \sum_{i,j=1}^J (T_i v, T_j v)_A \le K_1 (Tv, v)_A \le K_1 \|Tv\|_A \|v\|_A,$$

which implies (5.3.5).

If $v = \sum_{i=1}^J v_i$ is a decomposition that satisfies (5.3.2), then

$$(v, v)_A = \sum_{i=1}^J (v_i, v)_A = \sum_{i=1}^J (v_i, P_i v)_A,$$

and by the Cauchy-Schwarz inequality

$$\begin{aligned}\sum_{i=1}^J (v_i, P_i v)_A &= \sum_{i=1}^J (v_i, A_i P_i v) \le \sum_{i=1}^J (R_i^{-1} v_i, v_i)^{\frac{1}{2}} (R_i A_i P_i v, v)_A^{\frac{1}{2}} \\ &\le \ (\sum_{i=1}^J (R_i^{-1} v_i, v_i))^{\frac{1}{2}} (\sum_{i=1}^J (T_i v, v)_A)^{\frac{1}{2}} \le \sqrt{K_0} \|v\|_A (Tv, v)_A^{\frac{1}{2}}.\end{aligned}$$

Consequently

$$\|v\|_A^2 \le K_0(Tv, v)_A$$

which implies (5.3.6). □

As a consequence of (5.3.5), we have

Corollary 5.3.7 *A sufficient condition for Algorithm 5.2.8 to be convergent is that*

$$K_1 < 2.$$

(5.3.8) Remark. If K_0 is the smallest constant satisfying the inequality (5.3.2), then

$$\lambda_{\min}(BA) = K_0^{-1}.$$

In fact, for the trivial decomposition $v = \sum_{i=1}^J v_i$ with $v_i = T_i T^{-1} v$,

$$K_0 \le \max_{v \in \mathcal{V}} \frac{\sum_{i=1}^J (R_i^{-1} T_i T^{-1} v, T_i T^{-1} v)}{\|v\|_A^2} = \max_{v \in \mathcal{V}} \frac{(T^{-1}v, v)_A}{(v, v)_A} = (\lambda_{\min}(BA))^{-1}.$$

This together with (5.3.6) justifies our claim.

To present our next theorem, let us first prove a very simple but important lemma.

Lemma 5.3.9 *Denote, for $1 \le i \le J$, $E_i = (I - T_i)(I - T_{i-1}) \cdots (I - T_1)$ and $E_0 = I$. Then*

$$I - E_i = \sum_{j=1}^i T_j E_{j-1}, \tag{5.3.10}$$

$$(2 - \omega_1) \sum_{i=1}^J (T_i E_{i-1} v, E_{i-1} v)_A \le \|v\|_A^2 - \|E_J v\|_A^2, \quad \forall\, v \in \mathcal{V}. \tag{5.3.11}$$

Proof. (5.3.10) follows immediately from the trivial identity $E_{i-1} - E_i = T_i E_{i-1}$. From this identity, we further deduce that

$$\begin{aligned}
\|E_{i-1}v\|_A^2 - \|E_i v\|_A^2 &= \|T_i E_{i-1} v\|_A^2 + 2(T_i E_{i-1} v, E_i v)_A \\
&= (T_i E_{i-1} v, T_i E_{i-1} v)_A + 2(T_i (I - T_i) E_{i-1} v, E_{i-1} v)_A \\
&= ((2I - T_i) T_i E_{i-1} v, E_{i-1} v)_A \ge (2 - \omega_1)(T_i E_{i-1} v, E_{i-1} v)_A.
\end{aligned}$$

Summing up these inequalities with respect to i gives (5.3.11). □

Now we are in a position to present our second fundamental theorem.

Theorem 5.3.12 (Fundamental Theorem II) *For the Algorithm 5.2.13,*

$$\|E_J\|_A^2 \le 1 - \frac{2 - \omega_1}{K_0(1 + K_1)^2}, \tag{5.3.13}$$

where $\omega_1 = \max_i(R_i A_i)$.

Proof. In view of (5.3.1), (5.3.6) and (5.3.11), it suffices to show that

$$\sum_{i=1}^{J}(T_i v, v)_A \le (1+K_1)^2 \sum_{i=1}^{J}(T_i E_{i-1}v, E_{i-1}v)_A, \quad \forall\, v \in \mathcal{V}. \tag{5.3.14}$$

By (5.3.10)

$$\begin{aligned}(T_i v, v)_A &= (T_i v, E_{i-1}v)_A + (T_i v, (I - E_{i-1})v)_A \\ &= (T_i v, E_{i-1}v)_A + \sum_{j=1}^{i-1}(T_i v, T_j E_{j-1}v)_A.\end{aligned}$$

Applying the Cauchy-Schwarz inequality gives,

$$\sum_{i=1}^{J}(T_i v, E_{i-1}v)_A \le \left(\sum_{i=1}^{J}(T_i v, v)_A\right)^{\frac{1}{2}} \left(\sum_{i=1}^{J}(T_i E_{i-1}v, E_{i-1}v)_A\right)^{\frac{1}{2}}.$$

By the definition of K_1 in (5.3.3), we have

$$\sum_{i=1}^{J}\sum_{j=1}^{i-1}(T_i v, T_j E_{j-1}v)_A \le K_1 \left(\sum_{i=1}^{J}(T_i v, v)_A\right)^{\frac{1}{2}} \left(\sum_{j=1}^{J}(T_j E_{j-1}v, E_{j-1}v)_A\right)^{\frac{1}{2}}.$$

Combining these three formulae then leads to (5.3.14). □

This theorem shows that the SSC algorithm converges as long as $\omega_1 < 2$. The condition that $\omega_1 < 2$ is reminiscent of the restriction on the relaxation parameter in the SOR method. Since SOR (or block SOR) is a special SSC method, Theorem 5.3.12 gives another proof of its convergence for any symmetric positive definite system (and more details for this application will be given later).

5.3-b On the estimate of K_0 and K_1

Our fundamental theorems depend only on three parameters: ω_1, K_0 and K_1. Obviously there is little we can say about ω_1. We shall now discuss techniques for estimating K_0 and K_1.

We first state a simple result for the estimate of K_0.

Lemma 5.3.15 *Assume that, for any $v \in \mathcal{V}$, there is a decomposition $v = \sum_{i=1}^{J} v_i$ with $v_i \in \mathcal{V}_i$ satisfying*

$$\sum_{i=1}^{J}(v_i, v_i)_A \le C_0 (v, v)_A, \quad \textit{or} \quad \sum_{i=1}^{J}\lambda_i (v_i, v_i) \le \hat{C}_0 (v, v)_A,$$

where $\lambda_i = \rho(A_i)$; then $K_0 \le \frac{C_0}{\omega_0}$ or $K_0 \le \frac{\hat{C}_0}{\hat{\omega}_0}$ where

$$\omega_0 = \min_{1\le i\le J} \lambda_{\min}(R_i A_i) \quad \textit{and} \quad \hat{\omega}_0 = \min_{1\le i\le J} (\lambda_i \lambda_{\min}(R_i)).$$

The proof of the above lemma is straightforward.

We now turn to the estimate of K_1. For this purpose, we introduce a non-negative symmetric matrix $\mathcal{E} = (\epsilon_{ij}) \in \mathbb{R}^{J\times J}$ where ϵ_{ij} is the smallest constant satisfying

$$(5.3.16) \qquad (T_i u, T_j v)_A \le \omega_1 \epsilon_{ij} (T_i u, u)_A^{\frac{1}{2}} (T_j v, v)_A^{\frac{1}{2}}, \quad \forall\, u, v \in \mathcal{V}.$$

Clearly $\epsilon_{ij} \le 1$ and, $\epsilon_{ij} = 0$ if $P_i P_j = 0$. If $\epsilon_{ij} < 1$, the above inequality is often known as the *strengthened Cauchy-Schwarz inequality.*

Lemma 5.3.17

$$K_1 \le \omega_1 \rho(\mathcal{E}).$$

If $\epsilon_{ij} \lesssim \gamma^{|i-j|}$ *for some* $\gamma \in (0,1)$, *then* $\rho(\mathcal{E}) \lesssim (1-\gamma)^{-1}$ *and* $K_1 \lesssim \omega_1 (1-\gamma)^{-1}$. *In general* $\rho(\mathcal{E}) \le J$ *and* $K_1 \le \omega_1 J$.

Proof. Since $\mathcal{E}$ is symmetric, the estimate $K_1 \le \omega_1 \rho(\mathcal{E})$ follows by definition. The other estimates follow from the inequality $\rho(\mathcal{E}) \le \max_{1\le j\le J} \sum_{i=1}^{J} \epsilon_{ij}$. □
We shall now estimate K_1 in terms of $\mathcal{E}$ in a more precise fashion. To this end, we define, for a given subset $\mathcal{J}_0 \subset \{1, \cdots, J\}$,

$$\gamma_0 = |\mathcal{J}_0|, \quad \sigma_0 = \max_{j\notin \mathcal{J}_0} \sum_{i\notin \mathcal{J}_0} \epsilon_{ij}.$$

Here $|\mathcal{J}_0|$ denotes the number of elements in $\mathcal{J}_0$.

Lemma 5.3.18

$$K_1 \le \omega_1(\gamma_0 + \sigma_0).$$

Proof. Let

$$S_{11} = S\cap(\mathcal{J}_0\times\mathcal{J}_0), S_{12} = S\cap(\mathcal{J}_0\times\mathcal{J}_0^c), S_{21} = S\cap(\mathcal{J}_0^c\times\mathcal{J}_0), S_{22} = S\cap(\mathcal{J}_0^c\times\mathcal{J}_0^c).$$

Using (5.3.16), it is elementary to show that

$$(5.3.19) \quad \sum_{(i,j)\in S_{11}} (T_i u_i, T_j v_j)_A)^2 \le \omega_1^2 \gamma_0^2 \sum_{i\in\mathcal{J}_0} (T_i u_i, u_i)_A \sum_{j\in\mathcal{J}_0} (T_j v_j, v_j)_A,$$

$$(5.3.20) \quad \sum_{(i,j)\in S_{22}} (T_i u_i, T_j v_j)_A)^2 \le \omega_1^2 \sigma_0^2 \sum_{i\in\mathcal{J}_0^c} (T_i u_i, u_i)_A \sum_{j\in\mathcal{J}_0^c} (T_j v_j, v_j)_A.$$

Now, for any $i \in \mathcal{J}_0$, let $\mathcal{J}_0^c(i) = \{j \in \mathcal{J}_0^c : (i,j) \in S_{12}\}$. Then

$$\begin{aligned}
(\sum_{(i,j)\in S_{12}} (T_i u_i, T_j v_j)_A)^2 &= (\sum_{i\in\mathcal{J}_0} (T_i u_i, \sum_{j\in\mathcal{J}_0^c(i)} T_j v_j)_A)^2 \\
&\le \gamma_0 \sum_{i\in\mathcal{J}_0} \|T_i u_i\|_A^2 \| \sum_{j\in\mathcal{J}_0(i)} T_j v_j\|_A^2 \le \omega_1 \gamma_0 \sum_{i\in\mathcal{J}_0} (T_i u_i, u_i)_A \| \sum_{j\in\mathcal{J}_0(i)} T_j v_j\|_A^2
\end{aligned}$$

Similar to (5.3.20), for each i, we have

$$\| \sum_{j \in \mathcal{J}_0^c(i)} T_j v_j \|_A^2 \leq \omega_1 \sigma_0 \sum_{j \in \mathcal{J}_0^c} (T_j v_j, v_j)_A.$$

Consequently, we have shown that

$$(\sum_{(i,j) \in S_{12}} (T_i u_i, T_j v_j)_A)^2 \leq \omega_1^2 \gamma_0 \sigma_0 \sum_{i \in \mathcal{J}_0} (T_i u_i, u_i)_A \sum_{j \in \mathcal{J}_0^c} (T_j v_j, v_j)_A.$$

Similarly

$$(\sum_{(i,j) \in S_{21}} (T_i u_i, T_j v_j)_A)^2 \leq \omega_1^2 \gamma_0 \sigma_0 \sum_{i \in \mathcal{J}_0^c} (T_i u_i, u_i)_A \sum_{j \in \mathcal{J}_0} (T_j v_j, v_j)_A.$$

As $S = S_{11} \cup S_{12} \cup S_{21} \cup S_{22}$, the desired estimate follows by some elementary manipulations. $\square$

Chapter 6

One Dimensional Problems

In this chapter, we shall apply the theory above to one dimensional finite element methods. As the dicussions will be only confined in one dimensional space, everything is quite easy.

6.1 The finite element method

The finite element method for the approximation of (3.1.1) is based on an equivalent variational formulation of (3.1.1). To derive such a formulation, let us introduce the following functional space:

$$\mathcal{V} = \{v : \ v \text{ is continuous and piecewise differentiable}, v(0) = v(1) = 0\}.$$

Multiplying any function $v \in \mathcal{V}$ on both hand sides of (3.1.1) and integrating by parts, we get

$$\int_0^1 fv dx = -\int_0^1 u'' v dx = \int_0^1 u'v' dx. \tag{6.1.1}$$

Denote

$$A(u,v) = \int_0^1 u'v' dx.$$

We then have

$$A(u,v) = (f,v) \quad \forall\, v \in \mathcal{V}. \tag{6.1.2}$$

Like the finite difference discretization, a finite element discretization is also based on a partition of the interval $(0,1)$. Given any positive integer N, we consider the partition of the interval $(0,1)$ given by (3.1.2). Associated with the partition $\mathcal{T}_h$, we define a linear finite element space is defined as follows

$$\mathcal{V}_h = \{v : \ v \text{ is continuous and piecewise linear w.r.t. } \mathcal{T}_h,\, v(0) = v(1) = 0\}.$$

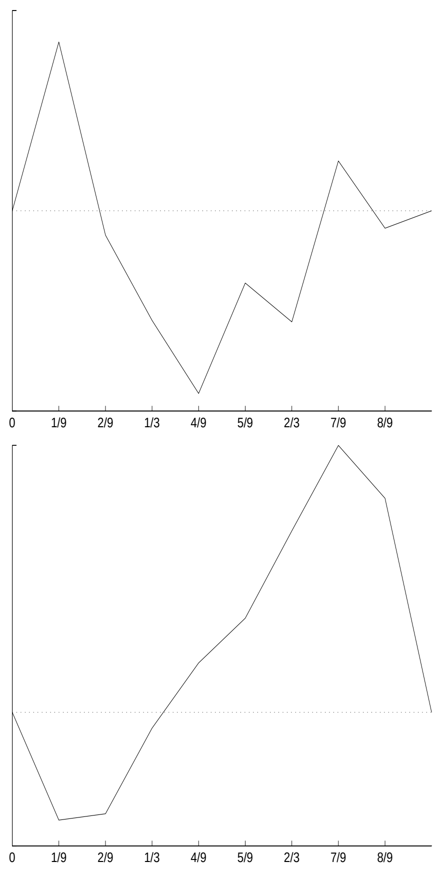


Figure 6.1: Typical finite element functions.

Namely $\mathcal{V}_h$ consists of piecewise linear continuous functions. An example function of $\mathcal{V}_h$ is shown in the Figure 6.1.

Apparently $\mathcal{V}_h$ is a linear vector space of dimension N and each internal nodal point corresponds to a degree of freedom. $\mathcal{V}_h$ has a natural basis, known as nodal basis, which is given by, for $i = 1, 2, \cdots, N$

$$\phi_i(x) = \begin{cases} \frac{x - x_{i-1}}{h}, & x \in [x_{i-1}, x_i]; \\ \frac{x_{i+1} - x}{h}, & x \in [x_i, x_{i+1}]; \\ 0 & \text{elsewhere.} \end{cases}$$

Note that $\phi_i(x_j) = \delta_{ij}$ and for any $v \in \mathcal{V}_h$,

$$v(x) = \sum_{i=1}^{N} v(x_i)\phi_i(x).$$

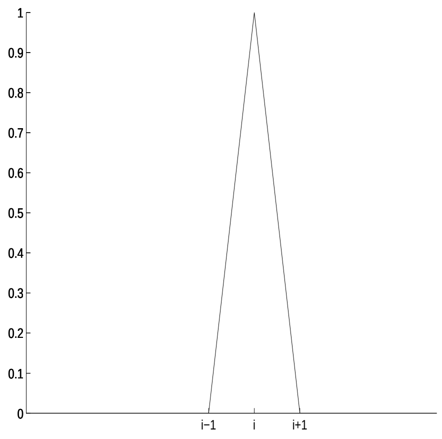


Figure 6.2: Finite element basis function ϕ_i.

The finite element approximation of (3.1.3) is defined as follows

$$\left\{ \begin{array}{l} \text{Find } u_h \in \mathcal{V}_h, \quad \text{such that} \\ A(u_h, v_h) = (f, v_h) \quad \forall\, v_h \in \mathcal{V}_h. \end{array} \right. \tag{6.1.3}$$

We write

$$u_h(x) = \sum_{i=1}^{N} \mu_i \phi_i(x) \text{ with } \mu_i = u_h(x_i).$$

The equation (6.1.3) is then equivalent to

$$\sum_{i=1}^{N} \mu_i A(\phi_i, \phi_j) = (f, \phi_j), \quad j = 1, 2, \cdots, N$$

or

$$A\mu = b. \tag{6.1.4}$$

Here $A = (A(\phi_i, \phi_j))_{N\times N}$ and $b = ((f, \phi_j))_{1\times N}$.

The matrix A is known as stiffness matrix under the nodal basis (ϕ_i).

A direct calculation shows that

$$\begin{aligned} \int_0^1 \phi'_{j-1}\phi'_j &= \int_0^1 \phi'_j \phi'_{j+1} = -\frac{1}{h}, \quad 1 < j < N \\ \int_0^1 (\phi'_j)^2 &= \frac{2}{h}, \quad 1 \le j \le N. \end{aligned}$$

Therefore, the finite element equation is reduced to

$$\frac{-\mu_{j-1} + 2\mu_j - \mu_{j+1}}{h} = f_j, \quad 1 \le j \le N, \tag{6.1.5}$$

where $f_j = \int_0^1 f\phi_j$.

We conclude that the stiffness matrix of the linear finite element approximation on a uniform grid, after proper scaling, is the same as the stiffness matrix of the finite difference scheme.

In this case, the stiffness matrix scaled by h is tridiagonal

$$A = \mathrm{diag}(-1, 2, -1)$$

and $\kappa(A) = O(h^{-2})$.

6.1-a Applications of Jacobi, GS and CG methods

As A is symmetric positive definite matrix, it is natural to apply Gauss-Seidel or conjugate gradient methods to solve (6.1.4). The convergence of these methods depends on the order of the matrix. It can be shown that

Proposition 6.1.6 *The convergence factor of Gauss-Seidel method for* (6.1.4) *is* $1 - ch^2$. *The convergence factor of conjugate gradient method for* (6.1.4) *is* $1 - ch$.

The convergence rate of CG method is obtained from the convergence estimate for CG method and the condition number estimate for A. The estimate for GS method will be seen late.

We observe that the convergence rates deteriorate as h decreases.

Nested multilevel finite element subspaces One convenient thing in finite element setting is that we can naturally talk about subspaces of piecewise linear functions. Associated with the multilevel grids $\mathcal{T}_k$ defined in §3.1-d, we can naturally define the following multilevel subspaces:

$$V_k = \{v : \ v \text{ is continuous and piecewise linear w.r.t. } \mathcal{T}_k,\ v(0) = v(1) = 0\}.$$

Since a piecewise linear function with respect to a coarse grid is naturally a piecewise linear function with respect to a finer grid, we have the following nested relation:

$$V_1 \subset V_2 \subset \ldots \subset V_J.$$

Let us now discuss how to restrict a fine grid finite element equation to a coarse grid finite element equation. Consider the following linear algebraic system on level k:

$$A_k \epsilon^k = \gamma^k.$$

This equation is equivalent to the following equation in variational form:

$$A(\epsilon^k, v_k) = (g, v_k) \quad \forall\, v_k \in V_k, \tag{6.1.7}$$

where

$$\epsilon^k(x) = \sum_{i=1}^{n_k} \epsilon_i \phi_i^k(x)$$

and g is the unique function in V_k satisfying

$$\gamma_i^k = (g, \phi_i^k).$$

The most natural restriction of the variational problem on V_{k-1} is the following

$$A(\epsilon^{k-1}, v_{k-1}) = (g, v_{k-1}) \quad \forall\, v_{k-1} \in V_{k-1}.$$

The algebraic system resulting from this is as follows:

$$A_{k-1}\epsilon^{k-1} = \gamma^{k-1},$$

where

$$\gamma_i^{k-1} = (g, \phi_i^{k-1}).$$

Using the following obvious relation

$$\phi_i^{k-1} = \frac{1}{2}\phi_{2i-1}^k + \phi_{2i}^k + \frac{1}{2}\phi_{2i+1}^k,$$

we then get

$$\gamma_i^{k-1} = \frac{1}{2}\gamma_{2i-1}^k + \gamma_{2i}^k + \frac{1}{2}\gamma_{2i+1}^k.$$

This is the exact relationship that defines the restriction matrix:

$$\gamma^{k-1} = I_k^{k-1}\gamma^k.$$

Let $I_k^{k-1} = (t_{ij}) \in R^{N_{k-1}\times N_k}$, then

$$\phi_i^{k-1} = \sum_{j=1}^{n_k} t_{ij}\phi_j^k.$$

Thus

$$a(\phi_i^{k-1}, \phi_j^{k-1}) = a(\sum_{l=1}^{n_k} t_{jl}\phi_l^k, \sum_{m=1}^{n_k} t_j m\phi_m^k) = \sum_{l,m=1}^{n_k} t_{jl}t_{jm}a(\phi_l^k, \phi_m^k),$$

which means

$$A_{k-1} = I_k^{k-1}A_k I_{k-1}^k.$$

6.1-b Nodal value interpolation

The interpolant is a linear operator $I_h : C[0,1] \mapsto \mathcal{V}_h$. For any $v \in C[0,1]$, $I_h v$ is the unique function in $\mathcal{V}_h$ satisfying

$$(I_h v)(x_i) = v(x_i), \quad 1 \le i \le N.$$

We need the following notations:

$$|v|_1 = (\int_0^1 (v')^2)^{1/2}, \quad |v|_2 = |v'|_1,$$

$$||v||_0 = (\int_0^1 v^2)^{1/2}, \quad ||v||_1 = (||v||_0^2 + |v|_1^2)^{1/2}.$$

Theorem 6.1.8 *For any* $v \in C^2[0,1] \cap C_0^1[0,1]$

$$\|v - I_h v\|_0 + h|v - I_h v|_1 \lesssim h^s |v|_{2-s}, \quad s = 0, 1. \tag{6.1.9}$$

In particular

$$\|I_h v\|_1 \lesssim \|v\|_1 \quad \forall\, v \in C_0^1[0,1]. \tag{6.1.10}$$

The above theorem is a direct consequence of the following local estimate.

Lemma 6.1.11 *For any* $1 \le i \le N+1$, $\tau_i = (x_{i-1}, x_i)$

$$\|v - I_h v\|_{0,\tau_i} \lesssim h|v|_{1,\tau_i}.$$

$$\|v - I_h v\|_{0,\tau_i} + h|v - I_h v|_{1,\tau_i} \lesssim h|v|_{2,\tau_i}.$$

Proof. Let $e(x) = (v - I_h v)(x)$, then $e(x_i) = e(x_{i-1}) = 0$ implies that there exists $\bar{x}_i \in [x_{i-1}, x_i]$ such that $e'(\bar{x}_i) = 0$. Thus, we have

$$e'(x) = \int_{\bar{x}_i}^{x} e''(x)dx = \int_{\bar{x}_i}^{x} v''dx,$$

and hence

$$|e''(x)| \le h^{1/2}(\int_{\tau_i} |v''|^2 dx)^{1/2}.$$

Integrating the above inequality on τ_i, we obtain

$$||v - I_h v||_{1,\tau_i} \lesssim h|v|_{2,\tau_i}.$$

Similarly, from the following identities

$$e(x) = \int_{x_{i-1}}^{x} e'(t)dt = \int_{x_{i-1}}^{x} \int_{\bar{x}_i}^{t} u''(s)ds,$$

$$e(x) = \lambda_1(x) \int_{x_{i-1}}^{x} v'(t)dt - \lambda_2(x) \int_{x}^{x_i} v'(t)dt,$$

we obtain other error estimates. Here

$$\lambda_1(x) = \frac{x_i - x}{h}, \quad \lambda_2(x) \frac{x - x_{i-1}}{h}.$$

□

6.1-c Error analysis

Lemma 6.1.12 *Assume that* u *and* u_h *are the solutions of* (3.1.3) *and* (6.1.3) *respectively then*

$$A(u - u_h, u - u_h) = \inf_{\chi \in \mathcal{V}_h} A(u - \chi, u - \chi)$$

and

$$\|u - u_h\|_1 \lesssim h|u|_2.$$

Proof. It is easy to see that

$$A(u,\chi) = (f,\chi) \quad \forall\, \chi \in V_h.$$

Thus

$$A(u-u_h,\chi) = 0 \quad \forall\, \chi \in V_h.$$

Consequently

$$\begin{aligned} A(u-u_h, u-u_h &= A(u-u_h, u-\chi) \\ &\leq A(u-u_h,u-u_h)^{\frac{1}{2}} A(u-\chi,u-\chi)^{\frac{1}{2}} \end{aligned}$$

which implies the first identity.

Taking $\chi = u_I$, the result then follows from (6.1.12) and (6.1.8). □

Introducing the Galerkin projection $P_h : \mathcal{V} \mapsto \mathcal{V}_h$ by

$$A(P_h v, \chi) = A(v,\chi) \quad \forall\, \chi \in \mathcal{V}_h.$$

Notice that the solution of (6.1.3) can be written as $u_h = P_h u$.

Lemma 6.1.13 *We have*

$$P_h = I_h.$$

More specifically

$$((u - I_h u)', \chi') = 0 \quad \forall\, u \in \mathcal{V}, \chi \in \mathcal{V}_h.$$

Proof. Note that χ' is piecewise constant, thus

$$((u-I_h u)', \chi') = \sum_{i=1}^{N+1} \chi'(u - I_h u)'(x) dx = 0.$$

□

6.2 Multilevel spaces

We assume the triangulation $\mathcal{T}$ is constructed by a successive refinement process. More precisely, $\mathcal{T} = \mathcal{T}_J$ for some $J > 1$ and $\mathcal{T}_k$, for $k \leq J$, are a nested sequence of quasi-uniform triangulations which consist of subintervals $\mathcal{T}_k = \{\tau_k^i\}$ of size h_k for $k = 1, \ldots, J$ such that $\Omega = \cup_i \tau_k^i$, where the quasi-uniformity constants are independent of k. These triangulations should be nested in the sense that any subinterval τ_{k-1}^l can be written as a union of subintervals of $\{\tau_k^i\}$. We further assume that there is a constant $\eta > 1$, independent of k, such that h_k is proportional to η^{-k}.

The simplest example is that $\mathcal{T}_k$ consists of the following internal nodes

$$x_i^k = \frac{i}{2^k}, \quad i = 1, 2, \cdots, n_k, k = 1, 2, \cdots, J$$

where $n_k = 2^k - 1$. Note that $\mathcal{N}_k$ is obtained by adding midpoints of the subintervals in $\mathcal{T}_{k-1}$. For each k the set of above nodes will be denoted by $\mathcal{N}_k$. In the rest of this section, these will be the multilevel grids to be used.

Corresponding to each triangulation $\mathcal{T}_k$, a finite element space $\mathcal{M}_k$ can be defined by $\mathcal{M}_k = \{v \in C_0^1(\Omega) : v|_\tau \in \mathcal{P}_1(\tau) \quad \forall\, \tau \in \mathcal{T}_k\}$, where $\mathcal{P}_1$ is the space of linear polynomials. Obviously

$$\mathcal{M}_1 \subset \mathcal{M}_2 \subset \cdots \subset \mathcal{M}_J \equiv \mathcal{M}. \tag{6.2.14}$$

In the discussion of multigrid iterative methods, we shall use the following Richardson iterator:

$$R_k v = \omega \sum_{i=1}^{n_k} (v, \phi_i^k)\phi_i^k \quad \forall\, v \in \mathcal{M}_k \tag{6.2.15}$$

where (ϕ_i^k) is the nodal basis of $\mathcal{M}_k$ and $\omega \in (0, 1/\rho(\mathcal{A}_k)$ with $\mathcal{A}_k$ being the stiffness matrix on $\mathcal{M}_k$.

We assume that $h = h_J$ is sufficiently small and h_1 is of unit size. Note that $J = O(|\log h|)$. For each k, we define the interpolant $I_k : C(\bar{\Omega}) \mapsto \mathcal{M}_k$ by

$$(I_k u)(x) = u(x) \quad \forall\, x \in \mathcal{N}_k.$$

Here $\mathcal{N}_k$ is the set of all nodes in $\mathcal{T}_k$. It follows from (6.1.9) and (6.1.10) that

$$\|(I_k - I_{k-1})v\|^2 + h_k^2 \|I_k v\|_A^2 \lesssim h_k^2 \|v\|_A^2, \quad v \in \mathcal{V}. \tag{6.2.16}$$

Lemma 6.2.17 *If $i \neq j$*

$$(((I_i - I_{i-1})v)', ((I_j - I_{j-1})v)') = 0.$$

Proof. By (6.1.13). □

(6.2.18) EXERCISE. Prove, for $i \le j$, that

$$\|I_i v\| \lesssim \sqrt{\frac{h_i}{h_j}} \|v\| \quad \forall\, v \in \mathcal{M}_j.$$

Lemma 6.2.19 *For all $v \in \mathcal{V}$*

$$\sum_{k=1}^{J} \|(I_k - I_{k-1})\|_1^2 \eqsim \|v\|_1^2.$$

Proof. Writing $v = \sum_k (I_k - I_{k-1})v$, we have, by (6.2.17)

$$\begin{aligned} |v|_1^2 &= \sum_{i,j=1}^{J} (((I_i - I_{i-1})v)', ((I_j - I_{j-1})v)') \\ &= \sum_{i}^{J} ((I_i - I_{i-1})v)', ((I_i - I_{i-1})v)'). \end{aligned}$$

As $|\cdot|_1 \eqsim \|\cdot\|_1$, the desired estimate then follows. □

6.2-a Strengthened Cauchy-Schwarz inequalities

This type of inequalities were used as assumptions in Chapter 5 (see (5.3.16)). Here we shall establish them for multilevel spaces.

Lemma 6.2.20 *Let $i \le j$; then*

$$A(u,v) \lesssim \gamma^{j-i} h_j^{-1} \|u\|_A \|v\| \quad \forall\, u \in \mathcal{M}_i, v \in \mathcal{M}_j. \tag{6.2.21}$$

Here, we recall, that $\gamma \in (0,1)$ is a constant such that $h_i \eqsim \gamma^{2i}$.

Proof. By Lemma (6.1.13), we have

$$A(u,v) = A(u, I_i v) \le |u|_1 |I_i v|_1.$$

It follows from inverse inequality and (6.2) that

$$|I_i v|_1 \lesssim h_i^{-1} \|I_i v\| \lesssim \frac{1}{\sqrt{h_i h_j}} \|v\|.$$

The estimate (6.2.21) then follows. □

Lemma 6.2.22 *Let $\mathcal{V}_i = (I_i - I_{i-1})\mathcal{V}$; then*

$$A(u,v) \lesssim \gamma^{|i-j|} \|u\|_A \|v\|_A \quad \forall\, u \in \mathcal{V}_i, v \in \mathcal{V}_j.$$

Proof. By (6.2.16), we have

$$\|v\| \lesssim h_i \|v\|_A \quad \forall\, v \in \mathcal{V}.$$

The result then follows directly from (6.2.20). □

Lemma 6.2.23 *Assume that $T_k = R_k A_k P_k$ and that $R_k : \mathcal{M}_k \mapsto \mathcal{M}_k$ satisfies*

$$\|R_k A_k v\|^2 \lesssim \lambda_k^{-1} (A_k v, v) \quad \forall\, v \in \mathcal{M}_k, \tag{6.2.24}$$

where $\lambda_k = \rho(A_k)$. Then for $1 \le i, j \le J$

$$(T_i u, T_j v)_A \lesssim \gamma^{\frac{|i-j|}{2}} (T_i u, u)^{\frac{1}{2}} (T_j v, v)^{\frac{1}{2}} \quad \forall\, u, v \in \mathcal{V}.$$

Proof. Assume that $i \le j$. An application of (6.2.20) yields

$$(u_i, T_j v)_A \lesssim \gamma^{j-i} h_j^{-1} \|u_i\|_A \|T_j v\|.$$

By (6.2.24)

$$\|T_j v\| = \|R_j A_j P_j v\| \lesssim h_j \|A_j^{\frac{1}{2}} P_j v\| \lesssim h_j \|v\|_A.$$

This proves that

$$(u_i, T_j v)_A \lesssim \gamma^{j-i} \|u_i\|_A \|v\|_A, \quad \forall\, u_i \in \mathcal{V}_i, v \in \mathcal{V}.$$

It follows from the Cauchy-Schwarz inequality and the inequality just proved:

$$\begin{aligned}(T_i u, T_j v)_A &\le (T_j v, v)_A^{\frac{1}{2}} (T_j T_i u, T_i u)_A^{\frac{1}{2}} \\ &\lesssim \gamma^{\frac{j-i}{2}} (T_j v, v)_A^{\frac{1}{2}} \|T_i u\|_A \lesssim \gamma^{\frac{j-i}{2}} (T_i u, u)_A^{\frac{1}{2}} (T_j v, v)_A^{\frac{1}{2}},\end{aligned}$$

as desired □

6.3 Hierarchical basis methods

Picture of three level HB basis functions

Lemma 6.3.25

$$(\psi_i', \psi_j') = \frac{2}{h_i}\delta_{ij}, \quad 1 \le i, j \le N.$$

Proof. By direct calculation or using the identity (6.2.17). □

(6.3.26) Remark. In fact $\{\psi_i'\}$, scaled by $\sqrt{h_i/2}$, is a subset of the well-known Haar basis (which is the socalled wavelets of the lowest order.)

Proposition 6.3.27 *The stiffness matrix under the hierarchal basis for problem* (6.1.3) *is diagonal.*

Preconditioning For the general equation (3.1.3), the property stated in (6.3.27) obviously is not valid any more. In fact, the corresponding stiffness matrix is dense. In this paragraph, we shall demonstrate how HB can be used in this more general and also more interesting case. The idea is to construct a preconditioner by means of the HB basis.

Theorem 6.3.28 *The condition number of the stiffness matrix $\hat{A}$ for* (3.1.3) *under the hierarchal basis is uniformly bounded.*

Proof. By Poincaré inequality,

$$|v|_1^2 \lesssim A(v, v) \lesssim |v|_1^2.$$

The desired result then follows from (6.3.27). □

As the HB stiffness matrix is well conditioned, CG method can then be applied. By (4.2.23), the convergence rate is uniform. However, such stiffness matrix is not sparse any more for general variable coefficients p and q and hence its action becomes expensive to compute.

To resolve this problem, we shall now establish a fact that $\hat{A}$ can be written as product of some sparse matrices.

Let $\{\phi_1^l, \phi_2^l, \cdots, \phi_{N_l}^l\}$ be the nodal basis functions of $V_l, N_l = 2^l - 1$, A be the NB stiffness matrix under $\{\phi_1^J, \phi_2^J, \cdots, \phi_{N_J}^J\} \equiv \{\phi_1, \phi_2, \cdots, \phi_{N_J}\}$. Define ψ_i^l by

$$\psi_1^1 = \phi_1^1, \quad \psi_i^l = \phi_{(2i-1)/2^l}^l, \quad i = 1, 2, \cdots, 2^{l-1} = N_{l-1} + 1,$$

then $\{\psi_1^l, \psi_2^l, \cdots, N_{l-1} + 1\}$ forms a basis for $V_l \setminus V_{l-1}$. If we use the notaion

$$\{\psi_1^1, \psi_1^2, \psi_2^2, \cdots, \psi_1^J, \cdots, \psi_{N_{J-1}+1}^J\} \equiv \{\psi_1, \cdots, \psi_{N_J}\}$$

and

$$\tilde{\psi}_i = (\frac{h_i}{2})^{1/2}\psi_i,$$

then

$$(\psi_i', \psi_j') = \frac{2}{h_i}\delta_{ij},$$

$$\hat{A} \equiv ((\tilde{\psi}_i', \tilde{\psi}_j')) = I_{N_J \times N_J},$$

where $I_{N_J \times N_J} \in R^{N_J \times N_J}$ is identical. Choose $R_l = \frac{h_l}{2} I_{(N_l - N_{l-1}) \times (N_l - N_{l-1})}$ and consider the algebraic representation of the PSC preconditioner

$$H = \sum_{l=1}^{J} S_l R_l S_l^t,$$

where $S_l \in R^{N_J \times (N_J - N_{J-1})}$ is the representation matrix of the basis

$$\{\psi_1^l, \psi_2^l \cdots, \psi_{N_{l-1}+1}^l\}$$

in terms of $\{\phi_1, \phi_2, \cdots, \phi_{N_J}\}$.

Note that for $S = ((\frac{h_1}{2})^{1/2} S_1, (\frac{h_2}{2})^{1/2} S_2, \cdots, (\frac{h_J}{2})^{1/2} S_J)$,

$$H = SS^t,$$

we get

$$\kappa(HA) = \kappa(SS^tA) = \kappa(S^tAS).$$

On the other hand,

$$(\tilde{\psi}_1, \tilde{\psi}_2, \cdots, \tilde{\psi}_{N_J}) = (\phi_1, \psi_2, \cdots, \phi_{N_J})S,$$

which implies

$$\begin{aligned} \hat{A} = ((\tilde{\psi}_i', \tilde{\psi}_j')) &= (\tilde{\psi}_1', \tilde{\psi}_2', \cdots, \tilde{\psi}_{N_J}')^t (\tilde{\psi}_1', \tilde{\psi}_2', \cdots, \tilde{\psi}_{N_J}') \\ &= ((\phi_1', \phi_2', \cdots, \phi_{N_J}')S)^t (\phi_1', \phi_2', \cdots, \phi_{N_J}')S \\ &= S^t((\phi_1', \phi_2', \cdots, \phi_{N_J}')^t (\phi_1', \phi_2', \cdots, \phi_{N_J}'))S \\ &= S^tAS. \end{aligned}$$

Therefore

$$\kappa(HA) = \kappa(\hat{A}) = 1.$$

Definition 6.3.29 $H = SS^t$ *is known as the hierarchal basis preconditioner.*

6.4 BPX preconditioners

Let $\mathcal{M}_k$ $(k = 1, \cdots, J)$ be the multilevel finite element spaces defined as in the preceding section. Again let $\mathcal{V} = \mathcal{M}_J$, but set $\mathcal{V}_k = \mathcal{M}_k$. In this case, the decomposition (5.1.1) is trivial.

We observe that, with $\mathcal{V}_k = \mathcal{M}_k$, there are redundant overlappings in the decomposition (5.1.1). The point is that these overlappings can be used advantageously to choose the subspace solvers in a simple fashion. Roughly speaking, the subspace solvers need only to take care of those "non-overlapped parts" (which correspond to the so-called *high frequencies*). Technically, the assumption on R_k is, for any $v \in \mathcal{M}_k$

$$\lambda_k{}^{-2}\|A_k v\|^2 \lesssim \|R_k A_k v\|^2 \le \omega_1 \lambda_k^{-1}(A_k v, v) \tag{6.4.30}$$

Note that (6.4.30) is a consequence of

$$\lambda_k{}^{-1}(v,v) \lesssim (R_k v, v) \le \omega_1 \lambda_k{}^{-1}(v,v), \quad \forall v \in \mathcal{M}_k$$

and it implies

$$\rho(R_k) \gtrsim \lambda_k^{-1} \quad \text{and} \quad \rho(R_k A_k) \le \omega_1. \tag{6.4.31}$$

All Richardson, damped Jacobi and (symmetric) Gauss-Seidel iteration satisfy this assumption. The proof for the Richardson or Jacobi method is straightforward. The proof for the Gauss-Seidel method is based on the following inequality

$$(A_k v, v) \le \frac{1}{2}(R_k^{-1} v, v), \quad \forall v \in \mathcal{M}_k$$

and the details id left for the reader.

Lemma 6.4.32 *Under the assumption* (6.4.30)

$$K_0 \lesssim 1 \quad \textit{and} \quad K_1 \lesssim 1.$$

Proof. By taking $v_i = (I_i - I_{i-1})v$ and applying (5.3.15), (6.2.19) and (6.4.31),

$$K_0 \lesssim 1.$$

The estimate for K_1 follows from (5.3.17) and (6.2.23). □

Combining (6.4.32) and the first fundamental theorem (5.3.4), we obtain

Theorem 6.4.33 *Assume that R_k's satisfy* (6.4.30)*; then the preconditioner* (5.2.7) *satisfies*

$$\kappa(BA) \lesssim 1.$$

Again, in view of (5.2.7), the algebraic representation of preconditioner (5.2.7) is

$$\mathcal{B} = \sum_{k=1}^{J} \mathcal{I}_k \mathcal{R}_k \mathcal{I}_k^t, \tag{6.4.34}$$

where $\mathcal{I}_k \in \mathbb{R}^{n\times n_k}$ is the representation matrix of the nodal basis $\{\phi_i^k\}$ in $\mathcal{M}_k$ in terms of the nodal basis $\{\phi_i\}$ of $\mathcal{M}$, i.e.

$$\Phi^k = \Phi \mathcal{I}_k$$

with $\Phi^k = (\phi_1^k, \cdots, \phi_{n_k}^k)$ and $\Phi = (\phi_1, \cdots, \phi_n)$.

Note that if we define $\mathcal{I}_k^{k+1} \in \mathbb{R}^{n_{k+1}\times n_k}$ such that

$$\Phi^k = \Phi^{k+1}\mathcal{I}_k^{k+1},$$

Then

$$\mathcal{I}_k = \mathcal{I}_{J-1}^J \cdots \mathcal{I}_{k+1}^{k+2}\mathcal{I}_k^{k+1}.$$

This identity is very useful for the efficient implementation of (6.4.34) on both serial and parallel fashions, cf. Xu and Qin [.xu qin.]. If $\mathcal{R}_k$ are given by the Richardson iteration, we have

$$\mathcal{B} = \sum_{k=1}^{J} h_k \mathcal{I}_k \mathcal{I}_k^t. \tag{6.4.35}$$

From (6.4.34) or (6.4.35), we see that the preconditioner depends entirely on the transformation between the nodal bases on multilevel spaces. For this reason, we shall call such kinds of preconditioners *multilevel nodal basis preconditioners.*

6.5 Domain decomposition with overlaps

Given an integer J, consider the following uniform partition of $(0,1)$:

$$0 = y_0 < y_1 < \cdots < y_{J+1} = 1, \quad y_j = \frac{j}{J+1}(j = 0 : J+1).$$

We assume that $\{y_j\} \subset \mathcal{N}_h$.

Based on this partition, a domain decomposition may be obtained as follows:

$$\Omega = \sum_{i=1}^{J} \Omega_i \quad \text{with} \quad \Omega_i = (y_{i-1}, y_i).$$

Based on these subdomains, the subspaces $\mathcal{V}_i$ $(1 \le i \le J)$ are defined by

$$\mathcal{V}_i = \{v \in \mathcal{V} : v(x) = 0 \quad \forall\, x \in \Omega \setminus \Omega_i\}.$$

If the number of subdomains J is too large, the above subspaces are not sufficient to produce an optimal algorithms. In regard to this consideration, we introduce a coarse finite element subspace $\mathcal{V}_0$ defined from the aforementioned partition of size $h_0 = 1/(J+1)$.

Lemma 6.5.36 *For the subspaces $\mathcal{V}_i$ $(0 \le i \le J)$, we have*

$$\mathcal{V} = \sum_{i=0}^{J} \mathcal{V}_i. \tag{6.5.37}$$

Furthermore there is a constant C_0 that is independent of h, h_0 or J, such that for any $v \in \mathcal{V}$, there are $v_i \in \mathcal{V}_i$ that satisfy $v = \sum_{i=0}^{J} v_i$ and

$$\sum_{i=0}^{J} a(v_i, v_i) \le C_0 a(v, v). \tag{6.5.38}$$

Proof. Let $I_0 : \mathcal{V} \mapsto \mathcal{V}_0$ be the nodal value interpolant. Given $v \in \mathcal{V}$, define $v_0 = I_0 v$,

$$v_1(x) = \begin{cases} (v - I_0 v)(x) & y_0 \le x \le y_1 \\ \frac{1}{2}(v - I_0 v)(x) & y_1 \le x \le y_0 \\ 0 & y_2 \le x \le 1, \end{cases}$$

for $j = 2 : J - 1$

$$v_j(x) = \begin{cases} \frac{1}{2}(v - I_0 v)(x) & y_{j-1} \le x \le y_j \\ 0 & \text{elsewhere} \end{cases}$$

and

$$v_J(x) = \begin{cases} 0 & 0 \le x \le y_{J-1} \\ \frac{1}{2}(v - I_0 v)(x) & y_{J-1} \le x \le y_J \\ (v - I_0 v)(x) & y_J \le x \le 1. \end{cases}$$

Obviously $v_i \in \mathcal{V}_i$ satisfying $v = \sum_{i=0}^{J} v_i$ and

$$\begin{aligned} \sum_{i=0}^{J} A(v_i, v_i) &\lesssim \|I_0 v\|_1^2 + \sum_{i=1}^{J} \|(v - I_0) v\|_{1,\Omega_i}^2 \\ &\lesssim \|I_0 v\|_1^2 + \|(v - I_0) v\|_1^2 \\ &\lesssim \|v\|_1^2. \end{aligned}$$

□

Lemma 6.5.39

$$K_0 \le C_0/\omega_0 \quad \text{and} \quad K_1 \le C.$$

Proof. The first estimate follows directly from (5.3.15) and (6.5.36). To prove the second estimate, we define

$$Z_i = \{1 \le j \le J : \Omega_i \cap \Omega_j \ne \emptyset\}.$$

By the construction of the domain decomposition, there exists a fixed integer n_0 such that

$$|Z_i| \le n_0 \quad \forall\, 1 \le i \le J.$$

Note that if $P_iP_j \neq 0$ or $P_jP_i \neq 0$, then $j \in Z_i$. An application of (5.3.18) (with $\mathcal{I}_0 = \{0\}$) gives $K_1 \leq \omega_1(1 + n_0)$. □

(6.5.40) Remark. An extreme case of a domain decomposition consists of the support of each nodal basis function:

$$\Omega_i = \text{supp } \{\phi_i\}.$$

In this case, the SSC method without using coarse grid space $\mathcal{V}_0$ is just the Gauss-Sedeil method.

A natural choice of a coarser space in this extreme case is a space, say $\mathcal{V}_{2h}$ defined on a partition with mesh size $2h$. The corresponding algorithm is a two level multigrid method. If this process is repeated to the space $\mathcal{V}_{2h}$ till a very coarse grid where an exact solver is used, the resultant algorithm is the multigrid method with Gauss-Seidel smoothers.

Chapter 7

An Introduction to Finite Element Methods

This chapter is to discuss the basic property of finite element approximation to second order elliptic equations, including the definition of finite element spaces, basic error estimates, basic adaptive techniques, property of finite element equations and applications of elementary iterative methods.

7.1 Ritz-Galerkin projection method

7.1-a Variational formulation of boundary value problems

We consider the following model problem:

$$-\Delta u = f$$

together with different boundary conditions.

Green's identities For a bounded Lipschitz domain Ω, the outer normal vector $\nu = (\nu_i)$ of $\partial\Omega$ can be well defined almost everywhere on $\partial\Omega$. The following identities will be frequently used in this book.

(7.1.1)
$$\int_\Omega (D_i u)v = -\int_\Omega u D_i v + \int_{\partial\Omega} uv\nu_i.$$

(7.1.2)
$$\int_\Omega \nabla u \cdot \nabla v = -\int_\Omega (\Delta u)v + \int_{\partial\Omega} \frac{\partial u}{\partial \nu} v.$$

(7.1.3)
$$\int_\Omega \text{div } w = \int_{\partial\Omega} w \cdot \nu$$

Dirichlet boundary condition From the Green's identities, one has that the Dirichlet boundary problem

$$\left\{ \begin{array}{rcll} -\Delta u & = & f, & \text{in}\Omega, \\ u & = & 0, & \text{on}\partial\Omega. \end{array} \right. \tag{7.1.4}$$

is equivalent to the variational formulation: Find $u \in H_0^1(\Omega)$ such that

$$a(u, v) = (f, v), \quad \forall v \in H_0^1(\Omega), \tag{7.1.5}$$

where

$$a(u, v) = \int_\Omega \nabla u \cdot \nabla v.$$

In fact, if $u \in H_0^1(\Omega)$ satisfying (7.1.5) is smooth, then from the Greend's identities, we get

$$\int_\Omega (\Delta u + f)v = 0, \quad \forall v \in H_0^1(\Omega),$$

which implies u satisfies (7.1.6).

Neumman boundary condition Similarly, one has that the Neumman boundary problem with the well-posed condition $\int_\Omega f = 0$

$$\left\{ \begin{array}{rcll} -\Delta u & = & f, & \text{in}\Omega, \\ \frac{\partial u}{\partial n} & = & 0, & \text{on}\partial\Omega. \end{array} \right. \tag{7.1.6}$$

is equivalent to the variational formulation: Find $u \in H^1(\Omega)$ such that

$$a(u, v) = (f, v), \quad \forall v \in H^1(\Omega). \tag{7.1.7}$$

7.1-b Ritz-Galerkin method

Let V_h be a finite dimensional subspace of continuous and piecewise smooth functions on Ω, V_h^0 is the subspace of V_h of vanishing on $\partial\Omega$. The so-called Ritz-Galerkin method for the Dirichelt boundary problem (7.1.6) is defined by: Find $u_h \in V_h^0$ such that

$$a(u_h, v) = (f, v), \quad \forall v \in V_h^0. \tag{7.1.8}$$

Theorem 7.1.9 *There holds*

$$\|u - u_h\|_a = \inf_{v \in V_h} \|u - v\|_a,$$

where

$$\| \cdot \|_a = (a(\cdot, \cdot))^{1/2}.$$

Proof. Frist, one sees that

$$a(u-u_h,v)=0, \quad \forall v\in V_h,$$

which leads to

$$\begin{aligned}\|u-u_h\|_a^2 &= a(u-u_h,u-u_h)\\ &= \inf_{v\in V_h} a(u-u_h,u-v)\\ &\le \|u-u_h\|_a \inf_{v\in V_h}\|u-v\|_a,\end{aligned}$$

or

$$\|u-u_h\|_a \le \inf_{v\in V_h}\|u-v\|_a.$$

This completes the proof.

Similarly, we can form a Ritz-Galerkin scheme and related error estimate for (7.1.7).

7.2 Linear finite element spaces

A finite element space is a finite dimensional vector space consisting of piecewise polynomials with respect to a partition of a domain.

7.2-a Triangulations

Given a bounded tetrahedral domain $\Omega\subset\mathbb{R}^d$. A triangulation of Ω is a special partition of Ω consisting of a set $\mathcal{T}_h$ of $d-$simplices with

$$h_\tau=\operatorname{diam}(\tau), h=\max_{\tau\in\mathcal{T}_h} h_\tau; \quad \underline{h}=\min_{\tau\in\mathcal{T}_h} h_\tau,$$

such that the intersection of any two simplices in $\mathcal{T}_h$ either consists of a common lower dimensional simplex or is empty and

$$\bar\Omega_h=\cup\{\bar\tau:\ \tau\in\mathcal{T}_h\}$$

and there exists a constant $\sigma_2>0$ satisfying

$$\underline{h}\lesssim h^{\sigma_2}.$$

The triangulation $\{\mathcal{T}_h: h\in\aleph\}$ is said to be *shape-regular* if

$$\sup_{h\in\aleph}\max_{\tau\in\mathcal{T}_h}\frac{h_\tau}{\rho_\tau}\le\sigma_1 \tag{7.2.1}$$

where ρ_τ denotes the radius of the ball inscribed in τ. The triangulation $\{\mathcal{T}_h: h\in\aleph\}$ is said to be *quasi-uniform* if it satisfies (7.2.1) and the following

$$h\le\sigma_3\underline{h}. \tag{7.2.2}$$

The assumption (7.2.1) is a local assumption, as is meant by above definition, for $d = 2$ for example, it assures that each triangle will not degenerate into a segment in the limiting case. A triangulation satisfying this assumption is often called to be *shape regular.*

On the other hand, the assumption (7.2.2) is a global assumption, which says that the smallest mesh size is not too small compared with the largest mesh size of the same triangulation. By the definition, in a quasiuniform triangulation, all the elements are about the same size asymptotically.

In the rest of this chapter, unless otherwise noted, we assume that the triangulation $\{\mathcal{T}_h : h \in \aleph\}$ is quasi-uniform.

Barycentric coordinates We note that an n-simplex τ in $\mathbb{R}^n$ is the convex hull τ of $n+1$ points $a_j = (a_{ij})_{i=1}^n \in \mathbb{R}^n$, which are called the vertices of the n-simplex, namely

$$\tau = \{x = \sum_{j=1}^{n+1} \lambda_j a_j; 0 \le \lambda_j \le 1, 1 \le j \le n+1, \sum_{j=1}^{n+1} \lambda_j = 1\}.$$

In the above, $\lambda_j = \lambda_j(x), 1 \le j \le n+1$, are called the barycentric coordinates of any point $x \in \tau$. It is easy to see that

$$\lambda_j(x) = |\tau_j(x)|/|\tau| \in P_1(\tau)$$

where $\tau_j(x)$ is the n-simplex with vertices x and all a_i except a_j, and $|\tau|$ is the measure of τ. One has

Lemma 7.2.3 *Given any linear function $v \in P_1(\tau)$. The v is uniquely determined by its values on the vertes of τ, namely $v(a_j), 1 \le j \le n+1$.*

Lemma 7.2.4 *The barycentric coordinates have the following properties*

1. $\lambda_i(a_j) = \delta_{ij}$.
2. $\sum_{j=1}^{n+1} \lambda_j(x) = 1$ *for all* $x \in \tau$.
3. $v(x) = \sum_{j=1}^{n+1} v_j \lambda_j(x)$ *for any* $v \in P_1(\tau)$.

7.2-b Finite element spaces

Corresponding to the triangulations $\mathcal{T}_h$, the finite element spaces $\mathcal{S}^h = \mathcal{S}^h(\Omega)$ is defined by

$$\mathcal{S}^h(\Omega) = \{v \in C(\bar{\Omega}) : v|_\tau \in \mathcal{P}_1(\tau), \quad \forall\, \tau \in \mathcal{T}_h\}.$$

It is easy to see that $\mathcal{S}^h(\Omega) \subset H^1(\Omega)$. We further define that

$$\mathcal{S}_0^h(\Omega) = \mathcal{S}^h(\Omega) \cap H_0^1(\Omega).$$

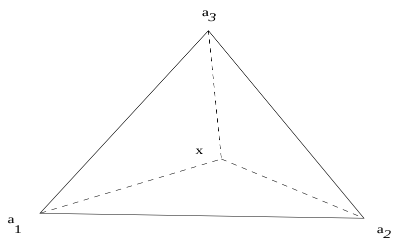


Figure 7.1: A 2-simplex (triangle)

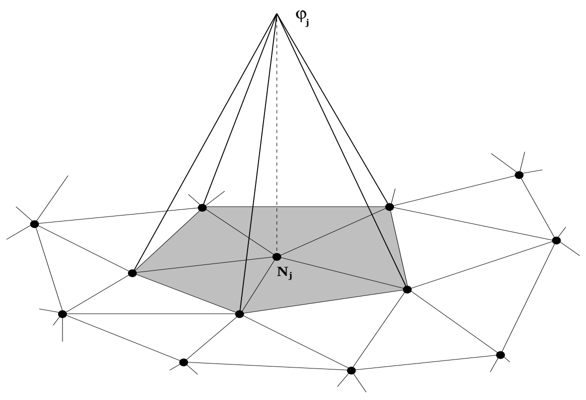


Figure 7.2: A typical nodal basis function

Nodal basis function Let $\mathcal{N}_h = \{x_i, i = 1, \dots, N_h\}$ be the set of all interior nodes of $\mathcal{T}_h$ and $\{\phi_1, \dots, \phi_{n_h}\}$ be the standard piecewise linear nodal basis functions such that ϕ_i is equal to one at precisely one node x_i and vanishs at all other nodal points. Namely

$$\phi_i(x_j) = \delta_{ij}.$$

It is clear that ϕ_i is locally supported. In fact

$$\text{supp } \phi_i = \bigcup_{\tau \in \mathcal{T}_h, x_i \in \tau} \tau.$$

Lemma 7.2.5 *For any $v \in \mathcal{S}^h$,*

$$v(x) = \sum_{j=1}^{N_h} v(x_j)\phi_j(x).$$

Proof. From Lemma 7.2.3, we immediately obtain the result. $\square$

7.2-c Nodal value interpolant

The nodal value interpolation operator $I_h : C(\bar{\Omega}) \mapsto \mathcal{S}^h$ is defined as follows

$$(I_h u)(x_i) = u(x_i), \quad \forall\, x_i \in \mathcal{N}_h.$$

The first result to be included is well-known and can be obtained by the Bramble-Hilbert Lemma.

Theorem 7.2.6 *Assume $d \le 3$ and $\{\mathcal{T}_h : h \in \aleph\}$ is quasiuniform, then*

$$\|(I - I_h)v\| + h\|(I - I_h)v\|_1 \lesssim h^2 |v|_2 \quad \forall\, v \in H^2(\Omega) \cap H_0^1(\Omega).$$

Proof. Let $x = (x^1, \dots, x^n)$ and $a_i = (a_i^1, \dots, a_i^n)$. Introducing the auxiliary functions

$$g_i(t) = v(a_i(t)), \text{ with } a_i(t) = a_i + t(x - a_i),$$

we have

$$g_i'(t) = (\nabla v)(a_i(t)) \cdot (x - a_i) = \sum_{l=1}^{n} (\partial_l v)(a_i(t))(x^l - a_i^l)$$

and

$$g_i''(t) = \sum_{k,l=1}^{n} (\partial_{kl}^2 v)(a_i(t))(x^k - a_i^k)(x^l - a_i^l). \tag{7.2.7}$$

By Taylor expansion

$$g_i(0) = g_i(1) - g_i'(1) + \int_0^1 t g_i''(t) dt.$$

Namely

$$v(a_i) = v(x) - (\nabla v)(x) \cdot (x - a_i) + \int_0^1 t g_i''(t) dt.$$

Note that

$$(I_h v)(x) = \sum_{i=1}^{n+1} v(a_i)\lambda_i(x), \sum_{i=1}^{n+1} \lambda_i(x) = 1,$$

and

$$\sum_{i=1}^{n+1} (x - a_i)\lambda_i(x) = 0.$$

It follows that

$$(I_h v - v)(x) = \sum_{i=1}^{n+1} \lambda_i(x) \int_0^1 t g_i''(t) dt \tag{7.2.8}$$

Using (7.2.7) and the trivial fact that $|x^l - a_i^l| \le h$, we obtain

$$\|g_i''(t)\|_{L^2(\tau)} \le h^2 \sum_{k,l=1}^{n} \|(\partial_{kl}^2 v)(a_i(t))\|_{L^2(\tau_i^t)} \le h^2 t^{-n/2} \sum_{k,l=1}^{n} \|\partial_{kl}^2 v\|_{L^2(\tau)}$$

where we have used the following change of variable

$$y = a_i + t(x - a_i) : \tau \mapsto \tau_i^t \subset \tau \text{ with } dy = t^n dx.$$

Now taking the $L^2(\tau)$ norm on both hand of sides of (7.2.8), we get

$$\begin{aligned}
\|I_h v - v\|_{L^2(\tau)} &\le h^2 \sum_{i=1}^{n+1} \max_{x\in\tau} |\lambda_i(x)| \int_0^1 t \|g_i''(t)\|_{L^2(\tau)}\, dt \\
&\le (n+1) \int_0^1 t^{1/n/2} dt\; h^2 \sum_{k,l=1}^{n} \|\partial_{kl}^2 v\|_{L^2(\tau)} \\
&\le \frac{2(n+1)}{4-n} h^2 \sum_{k,l=1}^{n} \|\partial_{kl}^2 v\|_{L^2(\tau)} \\
&\le \frac{4n(n+1)}{4-n} h^2 |v|_{H^2(\tau)}
\end{aligned}$$

Now we prove the H^1 error estimate. Taking the gradient on (7.2.8) and noticing that

$$\sum_{i=1}^{n+1} \lambda_i(x) \nabla \int_0^1 t g_i''(t) dt = \sum_{i=1}^{n+1} \lambda_i(x) \nabla \Big(v(a_i) - v(x) + (\nabla v)(x) \cdot (x - a_i) \Big) = 0,$$

we get

$$(\nabla(I_h v - v))(x) = \sum_{i=1}^{n+1} (\nabla \lambda_i)(x) \int_0^1 t g_i''(t) dt. \tag{7.2.9}$$

Then the estimate for $|\nabla(I_h v - v)|_{L^2(\tau)}$ follows by a similar argument and the following obvious estimate

$$|(\nabla \lambda_i)(x)| \lesssim \frac{1}{h}.$$

□

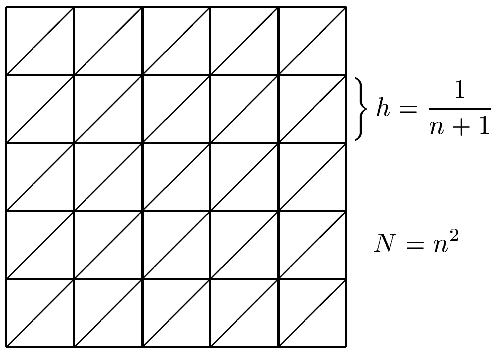
7.3 Finite element methods

With the finite element space $\mathcal{S}_0^h$ defined above, the finite element approximation of (**??**) is defined as follows:

$$u_h \in \mathcal{S}_0^h \quad a(u_h, \phi) = (f, \phi) \quad \forall\, \phi \in \mathcal{S}_0^h. \tag{7.3.10}$$

7.3-a A special case and relation to finite difference method

Let us first consider a special example in two dimensions. We consider a unit square and uniform triangulations:



Let $u_{i,j} = u_h(x_i, y_j)$. A direct calculation shows that the finite elemen equation (7.3.10) is reduced to

$$4u_{ij} - (u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}) = b_{i,j},$$

where

$$b_{i,j} = \int_\Omega f\phi_i\phi_j.$$

As a consequence, the stiffness matrix A is block tri-diagonal $A = \text{diag}\,(-I, B, -I)$ with $B = \text{diag}\,(-1, 4, -1)$.

For the five point stencil, we employ

$$\frac{\partial^2 u}{\partial x^2}(x_i, x_j) \approx \frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j)}{h^2} + O(h^2),$$

$$\frac{\partial^2 u}{\partial y^2}(x_i, x_j) \approx \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1})}{h^2} + O(h^2),$$

and

$$\Delta u \approx \frac{1}{h^2}(4u(x_i, y_j) - (u(x_{i+1}, y_j) + u(x_{i-1}, y_j) + u(x_i, y_{j+1}) + u(x_i, y_{j-1}))).$$

The finite difference scheme is then formed by

$$4u_{ij} - (u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}) = h^2 f_{i,j}.$$

Conclusion: In this special case, finite element and finite difference give rise to the same stiffness matrix.

7.3-b A basic error estimate

The basic error estimate is as follows, which is immediately obtained from Theorem 7.1.1 and Theorem 7.2.5

Theorem 7.3.11 *If the solution of* (??) $U \in H_0^1(\Omega) \cap H^2(\Omega)$, *then its finite element approximation* u_h *defined by* (7.3.10) *satisfies*

$$\|u - u_h\|_1 \lesssim h\|u\|_2.$$

7.4 Finite element matrices

In this section, we shall discuss some basic properties of the linear agebraic systems arising the discretization of elliptic boundary value problems.

7.4-a Basic properties

Let $\{\phi_j : 1 \le j \le N_h\}$ be the nodal basis functions. The the stiffness matrix $A = (a_{ij}), a_{ij} = a(\phi_i, \phi_j)$, has the following properties:

- A is sparse, has $O(N_h)$ nonzeros (since $a_{ij} = 0$ if x_i and x_j are not neighbors).
- A is SPD.

  In fact, A is obviously symmetric. Now, if $u_h = \sum_{j=1} N_h \mu_j \phi_j \in R^{N_h}$, then for $\mu = (\mu_1, \cdot, \mu_{N_h})$

$$\mu^t A \mu = \sum_{i,j=1}^{N_h} a_{ij}\mu_i\mu_j = \sum_{i,j=1}^{N_h} a(\phi_i, \phi_j)\mu_i\mu_j$$

$$= \quad a(u_h, u_h) \int_\Omega |\nabla u_h|^2 \ge 0.$$

7.4-b Conditioning of the stiffness matrix

The discrete elliptic operator A_h or the relevant stiffness matrix we have been studying is known to be ill-conditioned, namely $\kappa(A_h)$ grows very rapidly as the mesh size of the triangulation gets smaller. Some estimates for $\kappa(A_h)$ will be summarized below.

The first estimate for $\kappa(A_h)$ in the case of quasiuniform triangulation was given by Fried [?], who proved that $\kappa(A_h) \lesssim h^{-2}$. It was also mentioned (without proof however) by some authors (e.g. Axelsson and Barker [?]) that $\kappa(A_h) \gtrsim h^{-2}$. The following theorem includes such results.

Let us first consider a special example in two dimensions. We consider a unit square and uniform triangulations. Let $u_{i,j} = u(x_i, y_j)$. A direct calculation shows that the finite elemen equation (7.3.10) is reduced to

$$4u_{ij} - (u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}) = b_{i,j}.$$

As a consequence, the stiffness matrix A is block tri-diagonal $A = \text{diag}\,(-I, B, -I)$ with $B = \text{diag}\,(-1, 4, -1)$. By Example 1.5, the eigenvalues of A are as follows

$$\lambda_{ij}(A) = 4(\sin^2 \frac{i\pi}{2(N+1)} + \sin^2 \frac{j\pi}{2(N+1)}).$$

As a consequence, we have

$$\kappa(A) = \frac{\sin^2 \frac{N\pi}{2(N+1)} + \sin^2 \frac{N\pi}{2(N+1)}}{\sin \frac{2\pi}{2(N+1)} + \sin^2 \frac{\pi}{2(N+1)}},$$

which implies

$$\kappa(A) \eqsim N^2 \eqsim h^{-2}.$$

(7.4.1) EXERCISE. Prove that, with central finite difference methods with five point stencil, the resulting stiffness matrix, when properly scaled, is the same as the finite element discretization.

General cases

Theorem 7.4.2 *If the triangulatin is quasi-uniform, then we have for the stiffness matrix $\mathcal{A}$.*

$$\lambda_{\max}(\mathcal{A}) \eqsim h^{d-2}, \quad \lambda_{\min}(\mathcal{A}) \eqsim h^d, \quad \kappa(\mathcal{A}) \eqsim h^{-2}.$$

A conditioning estimate for nonuniform grids We shall now present a result due to Bank and Scott on the conditioning estimate for the stiffness matrix relevant to a shape-regular triangulation that may not be quasi-uniform. Roughly speaking, the stiffness matrix for a general shape-regular grid is almost as good as that for a quasiuniform grid after diagonal scaling.

Theorem 7.4.3 *Assume that the triangulations $\mathcal{T}_h$ satisfy* (**??**), *then we have*

$$\lambda_d^{-1}(h)\alpha^t\mathcal{D}\alpha \lesssim \alpha^t\mathcal{A}\alpha \lesssim \alpha^t\mathcal{D}\alpha \tag{7.4.4}$$

here

$$\lambda_{d,k} = \begin{cases} n_h^{-\frac{2}{d}}, & \textit{if } d \geq 3; \\ n_h^{-1}(1 + |\log h|)^{-1}, & \textit{if } d = 2. \end{cases}$$

Consequently

$$\kappa(\mathcal{D}^{-1}\mathcal{A}) \lesssim \kappa(A^k) = \begin{cases} n_h^{\frac{2}{d}}, & \textit{if } d \geq 3; \\ n_h(1 + |\log h|), & \textit{if } d = 2. \end{cases}$$

7.4-c Other cases

An example of finite element space is the linear element

$$\mathcal{S}^h(\Omega) = \{v \in C(\bar{\Omega}) : v \in P_1(\tau), \quad \forall \tau \in \mathcal{T}_h, v\mid_{\partial\Omega} = 0\}.$$

The higher order elements spaces are defined by

$$\mathcal{S}_r^h(\Omega) = \{v \in C(\bar{\Omega}) : v \in P_r(\tau), \quad \forall \tau \in \mathcal{T}_h, v\mid_{\Omega} = 0\}$$

and

$$\mathcal{S}_{r,0}^h(\Omega) = \{v \in \mathcal{S}_r^h(\Omega) : v\mid_{\partial\Omega} = 0\}.$$

Now we consider other boundary conditions. The Dirichlet boundary problem

$$\begin{cases} -\Delta u &= f, \quad \text{in}\Omega, \\ u &= u_D, \quad \text{on}\partial\Omega \end{cases} \tag{7.4.5}$$

is equivalent to the variational formulation: Find $u \in H^1(\Omega)$ such that $u\mid_{\partial\Omega} = u_D$ and

$$a(u,v) = (f,v), \quad \forall v \in H_0^1(\Omega). \tag{7.4.6}$$

The corresponding finite element approximation is: Find $u_h \in \mathcal{S}_r^h(\Omega)$ such that $u_h\mid_{\partial\Omega} = u_D^h$ and

$$a(u_h,v) = (f,v), \quad \forall v \in \mathcal{S}_{r,0}^h(\Omega), \tag{7.4.7}$$

where

$$u_D^h = \sum_{x_i \in \partial\Omega \cap \mathcal{N}_h} u_D(x_i)\phi_i(x).$$

Assume that

$$\int_\Omega f + \int_{\partial\Omega} g = 0,$$

and we consider the Neumann boundary problem

$$\begin{cases} -\Delta u &= f, \quad \text{in}\Omega, \\ \frac{\partial u}{\partial n} &= g, \quad \text{on}\partial\Omega, \end{cases} \tag{7.4.8}$$

which is equivalent to the variational formulation: Find $u \in H^1(\Omega)$ such that

$$a(u,v) = (f,v) + \int_{\partial\Omega} gv, \quad \forall v \in H_0^1(\Omega). \tag{7.4.9}$$

The corresponding finite element approximation is: Find $u_h \in \mathcal{S}_r^h(\Omega)$ such that

$$a(u_h,v) = (f,v) + \int_{\partial\Omega} gv, \quad \forall v \in \mathcal{S}_r^h(\Omega). \tag{7.4.10}$$

Similar error estimates can be obtained.

7.4-d M-matrix property

This kind of identities may be found in many papers, c.f. ... For completeness, let us now include a proof. Let λ_i $(1 \le i \le n+1)$ be the barycentric coordinate corresponding to the nodal point q_i, then, by definition, $a_{ij}^T = (\nabla\lambda_i, \nabla\lambda_j)$. Let F_i be the $n-1$ simplex opposite to a_i. Note that F_i is on the level set $\{x : \lambda_i(x) = 0\}$, thus the constant vector $\nabla\lambda_i$ is normal to F_i. Consequently $a_{ij}^T = -|T|\nabla\lambda_i \cdot \nabla\lambda_j = |T|\,|\nabla\lambda_i|\,|\nabla\lambda_j|\cos\theta_{\mathrm{E}}$, where $|\cdot|$ is the Euclidean norm. Let $\tilde{q}_i$ be the orthogonal projection of q_i onto F_i, then by Taylor expansion $0 = \lambda_i(\tilde{q}_i) = \lambda_i(q_i) + \nabla\lambda_i \cdot (\tilde{q}_i - q_i) = 1 - |\nabla\lambda_i| d_i$ where $d_i = \mathrm{dist}\ (q_i, F_i)$, thus $|\nabla\lambda_i| = d_i^{-1}$.

7.5 Some implementation issues

In this note, we discuss some details for the implementation of piecewise linear elements in two dimensions. Although, the technique to be described can be applied to more general problems, we will mainly focus on the Poisson equation in a polygonal domain Ω with Dirichlet boundary conditions and homogeneous Neumann conditions. The model differential equation we are dealing with is:

$$\begin{aligned} -\Delta u &= f(x,y), && (x,y) \in \Omega, \\ u &= u_D(x,y), && (x,y) \in \Gamma_D, \\ \frac{\partial u}{\partial \nu} &= 0, && (x,y) \in \Gamma_N. \end{aligned}$$

We triangulate Ω and we with n_{E} triangles and n number of nodes (vertices of the triangles) and as usual we denote the triangulation with $\mathcal{T}_h$.

7.5-a Basic arrays for defining the triangular grid

We define the basic arrays handling the triangulation. In what follows the dimension of the array will be in parenthesis after the array name. For example, an array named nt of dimension $3 \times n_{\mathrm{E}}$ will be denoted by $NT(3, n_{\mathrm{E}})$.

- n_{E} = number of elements
- n = number of nodes
- Integer array $IB(n,1)$ identifying the boundary nodes as follows:

$$IB(k) = \begin{cases} 0 & \text{if node } k \text{ is internal} \\ 1 & \text{if node } k \text{ is on } \Gamma_D \\ -1 & \text{if node } k \text{ is on } \Gamma_N \end{cases}$$

  We note that the nodes, where the boundary condition changes its type are assumed to be grid nodes. For example, if the condition changes from Dirichlet to Neumann at some point $(x,y) \in \partial\Omega$ then (x,y) is vertex of a triangle of our triangulation. It is important to note that in the actual computations such nodes are always treated as nodes on Dirichlet boundary (in some sense this matches with saying that Γ_D is a closed set in $\partial\Omega$ and Γ_N is open).

- Real array $xy(n,2)$ for the (x,y) coordinates of the nodes, namely (x_i,y_i) $i=1:n$.

- Integer array $NT(3,n_{\mathrm{E}})$ for the global numbering of three vertices of each element, namely a triangle numbered $\{ne\}$ has as a vertices

$$(NT(1,ne), NT(2,ne), NT(3,ne)).$$

  For example, $NT(3,6)=8$ means that the 6-th triangle has as a 3-rd vertex node with coordinates x_8, y_8.

7.5-b Local stiffness matrix

The implementation of finite element method is very easy to carry out because the computations can be done locally (element by element). To obtain the resulting system of discrete equations we only need to assemble all these local computations together. Roughly speaking, this is the main idea and we will follow it throughout the implementation.

First we have to compute the local stiffness matrices A_{T}. To do this we shall use the so-called *reference element*. The reference element is the triangle $\hat{\tau}$ with vertices $\hat{a}_1=(1,0)$, $\hat{a}_2=(0,1)$, $\hat{a}_3=(0,0)$. The advantage of this approach is that the integrals we need to evaluate, can be computed on the reference triangle (only once!). Then we use these values many times in obtaining different local stiffness matrices for the elements on the actual grid. In other words, if we want to compute an integral on some triangle $\tau\in\mathcal{T}_h$, we only need to evaluate the *affine mapping* which maps the reference triangle $\hat{\tau}$ on τ. This procedure is described in more detail below.

- Affine mapping:

  Assume $\tau\in\mathcal{T}_h$ is the $ne-$th element. Let

$$\{i,j,k\}=\{1,2,3\},\quad ie=NT(i,ne), je=NT(j,ne), ke=NT(k,ne).$$

  We define an affine mapping $F(\hat{x})=B\hat{x}+c:\hat{\tau}\mapsto\tau$:

$$B=\begin{pmatrix} x(ie)-x(ke) & x(je)-x(ke)\\ y(ie)-y(ke) & y(je)-y(ke)\end{pmatrix}$$

  and

$$c=\begin{pmatrix} x(ke)\\ y(ke)\end{pmatrix}.$$

- Real array $A_{\mathrm{T}}(3,3)$ for local stiffness matrix. If we take $ne-$th element τ then the local stiffness matrix is defined as follows.

$$A_{\mathrm{T}}(i,j) = a(\lambda_i, \lambda_j),$$

where λ_i are so called *barycentric coordinates*:

$$\lambda_i(x,y) = (\det(B))^{-1} \det \begin{pmatrix} x(je)-x & x(je)-x \\ y(ke)-y & y(ke)-y \end{pmatrix}$$

- Now we form the local stiffness matrix using reference element.

$$A_{\mathrm{T}}(i,j) = (\nabla\lambda_i, \nabla\lambda_j) = \frac{1}{2}\Big|\det(B)\Big|(B^{-1}\nabla\hat{\lambda}_i, B^{-1}\nabla\hat{\lambda}_j)$$

Here
$$\hat{\lambda}_1 = \hat{x}, \hat{\lambda}_2 = \hat{y}, \hat{\lambda}_3 = 1 - \hat{x} - \hat{y}.$$

7.5-c Assembling the global stiffness matrix and the right hand side

The main goal in doing all these computations is to form the resulting system of linear equation of the form:
$$Au = f$$

Here $A = A(n,n)$ is the global stiffness matrix and $f(n)$ is the right hand side and after solving this system, the vector $u(n)$ will contain the nodal values of the discrete solution.

The global stiffness matrix is assembled from the local ones as follows: For every particular element we first compute the local stiffness matrix $A_{\mathrm{T}}(3,3)$ and local right hand side $f_{\mathrm{T}}(3)$. Then we add them in $A(n,n)$, $f(n)$ respectively. The place, where to add them is uniquely determined by the local-global correspondence we have stored in $NT(3, n_{\mathrm{E}})$. Thus, for the local matrices and right hand sides only two small arrays $A_{\mathrm{T}}(3,3)$ and $f_{\mathrm{T}}(3)$ are used (but in the values in them are changed). More precisely the procedure is as follows:

1. Initializing A and b by taking the Dirichlet condition into account:

   - $A \leftarrow 0$, $f \leftarrow 0$
   - For $k = 1 : n$, if the node k is on the Dirichlet boundary (namely $IB(k) > 0$), then

   $$A(k,k) = 1, f(k) = u_D(x_k, y_k), \text{ the Dirichlet value at node } k.$$

2. Assembling

   For $ne = 1, \ldots n_{\mathrm{E}}$, do

- Form the local stiffness matrix $A_{\mathrm{T}}(3,3)$ for element ne (denoted by τ_{ne})
- For $i = 1,2,3$, if $NT(i,ne)$ is not a Dirichlet node, namely

  $$IB(NT(i,ne)) \leq 0, \text{ (with } ie = NT(i,ne), je = NT(j,ne)),$$

  $$f(ie) \leftarrow f(ie) + f_E$$

  where

  $$\begin{aligned} f_E &= \int_{\tau_{ne}} f(x)\lambda_i(x)\,dx \\ &= |\det(B)| \int_{\hat{\tau}} \hat{f}(\hat{x})\hat{\lambda}_i(\hat{x})\,d\hat{x} \\ &\approx |\det(B)| \frac{1}{12}\left(\hat{f}(\frac{\hat{a}_i + \hat{a}_j}{2}) + \hat{f}(\frac{\hat{a}_i + \hat{a}_k}{2})\right) \end{aligned}$$

  and, for $j = 1,2,3$, and if je is not a Dirichlet node (namely $IB(je) \leq 0$), do,

  $$A(ie,je) \leftarrow A(ie,je) + A_{\mathrm{T}}(i,j),$$

  otherwise

  $$f(je) \leftarrow f(je) - A_{\mathrm{T}}(i,j) * f(ie).$$

In the procedure described above a special care is taken for the Dirichlet values. It has to be noted that the right hand side for a Dirichlet node always must contain the corresponding Dirichlet value (the value of u_D). The right hand sides of the immediate neighbors to such boundary node have to be updated in proper way. We will explain this in some more detail. Let us pick a node k lying on the Dirichlet boundary and let j be in the interior of Ω and $A(j,k) \neq 0$. Then the j-th equation in our resulting system looks like:

$$A(1,j)u(1) + \ldots + A(k,j)u(k) + \ldots = f(j).$$

The known value of $u(k) = u_D(x_k, y_k)$ has to be moved to the right hand side, i.e.

$$A(1,j)u(1) + \ldots + 0 \cdot u(k) + \ldots = f(j) - A(k,j)u_D(x_k, y_k).$$

As it is seen, we are able to do this procedure locally (in one and the same loop with the assembling), because $A(j,k) \neq 0$ only if k-th and j-th node are in one and the same element.

To conclude, we would like to mention two immediate generalizations of the implementation we have just described. Although the differential equation considered here was fairly simple, the implementation can be extended very easy to handle more complicated equations and more complicated finite element methods. For example quadrature formulae for the evaluation of the integrals can be easily included if the simple model problem we consider is replaced by some more general elliptic PDE with non-constant coefficients. Another possible generalization is the use of higher order finite elements. In both cases only the computation of the integrals in the reference element has to be changed.

7.6 A posteriori error estimates and adaptivity

Theorem 7.6.1 *For the Poisson equation, there esists a constant C_0 only depending on the shape of triangulations such that,*

$$\|\nabla(u-u_h)\| \le C_0\|hR(u_h)\|,$$

where $R(u_h)=R_0(u_h)+R_1(u_h)$ and, with $\tau\in\mathcal{T}_h$

$$R_0(u_h)=|f+\Delta u_h|$$

and

$$R_1(u_h)=\frac{1}{2}\max_{S\subset\partial\tau} h_S^{-1}\left|\left[\frac{\partial u_h}{\partial n_\tau}\right]\right|.$$

Proof. We need to estimate $\|\nabla e\|$, where $e=u-u_h$. We know

$$(\nabla e,\nabla v)=0,\quad \forall v\in\mathcal{S}^h_{r,0}(\Omega),$$

thus, for any $v\in\mathcal{S}^h_{r,0}(\Omega)$

$$\begin{aligned}
&\|\nabla e\|^2=(\nabla e,\nabla e)-(\nabla e,\nabla v)=(\nabla e,\nabla(e-v))\\
\le\;& \sum_{\tau\in\mathcal{T}_h}\int_\tau(\nabla e,\nabla(e-v)=\sum_{\tau\in\mathcal{T}_h}\int_\tau -\Delta e(e-v)+\int_{\partial\tau}\frac{\partial u_h}{\partial n_\tau}(e-v)\\
=\;& (f+\Delta u_h,e-v)+\sum_{\tau\in\mathcal{T}_h}\int_{\partial\tau}\frac{\partial u_h}{\partial n_\tau}(e-v)\\
\equiv\;& E_1+E_2.
\end{aligned}$$

By the following trace theorem

$$\|w\|_{0,\partial\tau}\le C(h^{-1/2\|}w\|_{0,\tau}+h^{-1/2}\|\nabla w\|_{0,\tau},\quad \forall w\in H^1(\tau),$$

we get

$$\begin{aligned}
&\int_\tau[\frac{\partial u_h}{\partial n_\tau}](e-v)\le C\|\frac{\partial u_h}{\partial n_\tau}]\|_{0,\partial\tau}\|e-v\|_{0,\partial\tau}\\
\le\;& C\|hR_1(u_h)\|_{0,\tau}(h^{-1}\|e-v\|_{0,\tau}+\|\nabla(e-v)\|_{0,\tau}),
\end{aligned}$$

and hence

$$\begin{aligned}
&E_2=\sum_\tau\int_{\partial\tau}\frac{\partial u_h}{\partial n_\tau}(e-v)\\
=\;& \sum_{\tau\in\mathcal{T}_h}\sum_{s=\tau\cap\tau'}\int_s(\frac{\partial u_h}{\partial n_\tau}\mid_\tau-\frac{\partial u_h}{\partial n_\tau}\mid_{\tau'})(e-v)\\
=\;& \sum_{\tau\in\mathcal{T}_h}[\frac{\partial u_h}{\partial n_\tau}](e-v)\\
\le\;& C\|hR_1(u_h)\|(h^{-1}\|e-v\|+\|\nabla(e-v)\|).
\end{aligned}$$

Thus from the error estimate

$$\inf_{v\mathcal{S}^h_{r,0}(\Omega)} (\|e-v\| + h\|\nabla(e-v)\|) \le C\|\nabla e\|$$

and

$$|E_1| \le \|R_0(u_h)\|\|e-v\|,$$

we complete the proof.

We note that the total residual $R(u_h)$ has one contribution $R_0(u_h)$ from the interior of the elements τ, and one contribution from the element sides S involving the jump in the normal derivatives of u_h divided by the local mesh size. We note that the constant C_0 in the estimate above only depends on the shape of the element (more specifically, the lower bound of minimal angle). In principle, this constant can be estimated explicitly.

(7.6.2) Remark. We also like to remark, the mesh size h in the above estimate is viewed as a function of x. Thus it appears inside of the norm.

(7.6.3) Remark. For linear finite elements, we have of course $\Delta u_h = 0$ on each element.

Optimal shape-regular grid with fixed number of elements In the discussion of local mesh refinement, the mesh parameter h can not properly defined globably and for this reason we use an alternate notation for u_h, namely $u_{\mathcal{T}}$.

Given a shape-regular grid $\mathcal{T}$, let $h(x)$ denote the mesh-size function whose value is the mesh size of the element containing x.

Note that

$$\operatorname{card}(\mathcal{T}) = \sum_{\tau\in\mathcal{T}} 1 \approx \sum_{\tau\in\mathcal{T}} \int_\tau h^{-2}(x)dx = \int_\Omega h^{-2}(x)dx$$

and

$$\begin{aligned} \|u-u_{\mathcal{T}}\|^2_{1,\Omega} &\approx \|u-u_I\|^2_{1,\Omega} \approx \sum_{\tau\in\mathcal{T}} h(\tau)^2|u|^2_{2,\tau} \\ &\approx \sum_{\tau\in\mathcal{T}} \int_\tau h^2(x)|\nabla^2 u(x)|^2 dx \qquad (7.6.4) \\ &\approx \int_\Omega h^2(x)w(x)dx \end{aligned}$$

where $w(x) = |\nabla^2 u(x)|^2 \ge 0$.

Let N be a given positive (large) integer. We would like to seek for "optimal" grid with N elements in the sense that

$$\min \int_\Omega h^2(x)w(x)dx \qquad \text{subject to} \qquad \int_\Omega h^{-2}(x)dx = N.$$

The above constrained optimization problem may be solved by using Lagrangian multiplier method by considering the following nonlinear functional

$$L(h,\lambda)=\int_\Omega h^2(x)w(x)dx+\lambda\Big(\int_\Omega h^{-2}(x)dx-N\Big)$$

for

$$h\in L^2(\Omega),\lambda\in R.$$

The critical point of the above functional satisfies

$$2\int_\Omega h(x)\phi(x)w(x)dx-2\lambda\int_\Omega h^{-3}(x)\phi(x)dx=0\quad\forall\phi\in L^2(\Omega)$$

which implies

$$h(x)w(x)=\lambda h^{-3}(x)$$

namely, the optimal $h(x)$ should satisfies the following condition

$$h^4(x)w(x)=\lambda\quad\forall\, x\in\Omega. \tag{7.6.5}$$

Recall

$$\|u-u_{\mathcal{T}}\|_{1,\Omega}^2\approx\sum_{\tau\in\mathcal{T}}h(\tau)^2|u|_{2,\tau}^2\approx\sum_{\tau\in\mathcal{T}}h^4(x_\tau)w(x_\tau)$$

where for each τ we have used the mean value theorem with some $x_\tau\in\tau$. The above relation, in view of (7.6.5), means that the energy norm error is pretty much equally distributed over each element.

By (7.6.5), we have

$$N=\int_\Omega\frac{\sqrt{w(x)}}{\sqrt{\lambda}}dx,\quad\text{namely }\sqrt{\lambda}=N^{-1}\int_\Omega\sqrt{w(x)}dx.$$

Thus,

$$\|u-u_{\mathcal{T}}\|_{1,\Omega}^2\approx\int_\Omega h^2(x)w(x)dx=\int_\Omega\sqrt{w(x)}\sqrt{\lambda}dx,=N^{-1}\Big(\int_\Omega\sqrt{w(x)}dx\Big)^2.$$

In summary, loosely speaking, we have the following

Optimal-grid-criteria: For an optimal grid, the error in energy norm should be equally distributed over each element. More precisely, if $(\eta_\tau^{\mathcal{T}})^2$ is the distribution of $\|u-u_{\mathcal{T}}\|_{1,\Omega}^2$ over τ, namely

$$\|u-u_{\mathcal{T}}\|_{1,\Omega}^2\approx\sum_{\tau\in\mathcal{T}}(\eta_\tau^{\mathcal{T}})^2,$$

then

$$(\eta_\tau^{\mathcal{T}})^2\approx(\bar\eta^{\mathcal{T}})^2\equiv\frac{1}{\operatorname{card}(\mathcal{T})}\sum_{\tau\in\mathcal{T}}(\eta_\tau^{\mathcal{T}}).$$

and

$$\|u-u_{\mathcal{T}}\|_{1,\Omega}^2\approx\Big(\operatorname{card}(\mathcal{T})\Big)^{-1}.$$

An adaptive algorithm With the a posteriori error estimate given above, an adaptive algorithm can be obtained for automatic control of the energy norm error so that the error $\|\nabla(u-u_{\mathcal{T}})\|^2$ is guarenteed to satisfy for a given tolerance.

By ignoring the constant C_0, let us write the error estimator in the following form

$$\|u-u_{\mathcal{T}}\|_{1,\Omega}^2 \approx \sum_{\tau\in\mathcal{T}} (\eta_\tau^{\mathcal{T}})^2 \tag{7.6.6}$$

where $(\eta_\tau^{\mathcal{T}})^2 = \|hR(u_h)\|_{0,\tau}^2$.

Let tol is given tolerance, we need to adaptively find a grid $\mathcal{T}$ so that

$$\sum_{\tau\in\mathcal{T}} (\eta_\tau^{\mathcal{T}})^2 \le \text{tol}. \tag{7.6.7}$$

If (7.6.7) is not satisfied for the grid $\mathcal{T}'$, we need to refine the grid and obtain a new grid $\mathcal{T}'$ so that (**??**) may be valid for this new grid. In practice, it is desirable to refine more than a few elements of $\mathcal{T}$ before evaluate the error estimators again. Let N and N' denote the number of elements in $\mathcal{T}$ and $\mathcal{T}'$ respectively, one strategy is to assure that $N' = m_r N$ where $m_r > 1$ is given in advance. Typically, we choose $m_r = 2$ or $m_r = 4$.

We first note

$$\|u-u_{\mathcal{T}}\|_{1,\Omega}^2 = O((\text{card}(\mathcal{T}))^{-1}).$$

Assume that the exact solution $u \in H^2(\Omega)$, then

$$\|u-u_{\mathcal{T}}\|_{1,\Omega}^2 \approx \|u-u_I\|_{1,\Omega}^2 \approx \sum_{\tau\in\mathcal{T}} h(\tau)^2|u|_{2,\tau}^2 = \sum_{\tau\in\mathcal{T}} O(h(\tau)^4).$$

$$\frac{\sum\limits_{\tau\in\mathcal{T}'} (\eta_\tau^{\mathcal{T}'})^2}{\sum\limits_{\tau\in\mathcal{T}} (\eta_\tau^{\mathcal{T}})^2} \approx \frac{\|u-u_{\mathcal{T}}\|_{1,\Omega}^2}{\|u-u_{\mathcal{T}'}\|_{1,\Omega}^2} \approx \frac{(m_rN)^{-1}}{N^{-1}} = m_r^{-2}.$$

Here $\eta_\tau^{\mathcal{T}}$ and $\eta_\tau^{\mathcal{T}'}$ are the residual error estimates for an arbitrary triangle τ in $\mathcal{T}$ and $\mathcal{T}'$ respectively.

Assume that the error estimators are approximately equidistributed on $\mathcal{T}'$ so that $\eta_\tau^{\mathcal{T}'} \approx \bar{\eta}^{\mathcal{T}'}$ for all $\tau \in \mathcal{T}'$, we obtain

$$\bar{\eta}^{\mathcal{T}'} = \frac{m_r^{-3}}{\text{card}(\mathcal{T})} \sum_{\tau\in\mathcal{T}} (\eta_\tau^{\mathcal{T}})^2.$$

Then by comparing the last two terms, we get

$$(\eta_\tau^{\mathcal{T}})^2 \approx h(\tau)^4.$$

When a triangle τ is bisected into τ_1 and τ_2, we have

$$h(\tau_i)^2 \approx \text{area}(\tau_i) = \frac{1}{2}\text{area}(\tau) \approx \frac{1}{2}h(\tau)^2.$$

Assuming that $\eta_{\tau_i}^{\mathcal{T}'}$ are approximately equal, it follows that

$$\frac{(\eta_{\tau_i}^{\mathcal{T}'})^2}{(\eta_{\tau}^{\mathcal{T}})^2} \approx \frac{h(\tau_i)^4}{h(\tau)^4} = \frac{1}{4}.$$

By a similar argument, if τ_i is any descendent of τ obtained by applying g_τ times, then

$$\frac{(\eta_{\tau_i}^{\mathcal{T}'})^2}{(\eta_{\tau}^{\mathcal{T}})^2} \approx \frac{h(\tau_i)^4}{h(\tau)^4} = \frac{1}{4^{g_\tau}}.$$

The equation for determining g_τ is

$$4^{-g_\tau}(\eta_\tau^{\mathcal{T}})^2 = (\bar{\eta}^{\mathcal{T}'})^2$$

or

$$g_\tau = \frac{\log(\eta_\tau^{\mathcal{T}}/\bar{\eta}^{\mathcal{T}'})}{\log 2}.$$

1. Choose an initial mesh $\mathcal{T}_0$ of mesh size h_0.
2. Compute the corresponding finite element solution $u_0 \in V_0$ defined on $\mathcal{T}_0$.
3. Given a computed solution $u_j \in V_j$ on a mesh $\mathcal{T}_j$ with mesh size h_j, stop if

   (7.6.8) $$C_0\|h_j R(u_j)\| \le \text{tol}.$$

4. If not, try to determine a new mesh T_{j+1} with mesh function h_{j+1} with maximal possible size such that

   (7.6.9) $$C_0\|h_{j+1} R(u_{j+1})\| = \text{tol}.$$

   More specifically, the estimate 7.6.9 is attempted by equally distributing the error on each element on the new mesh which is obtained by subdividing elements with large contribution to the error:

   (7.6.10) $$C_0^2\|(h_\tau R(u_j)|_\tau)^2 h_\tau^n = \frac{\text{tol}^2}{N_{j+1}} \quad \forall\, \tau \in \mathcal{T}_{j+1}$$

   where $R(u_{j+1})|_\tau$ is the maximum value of the resdual $R(u_j)$ on the element τ and N_{j+1} is the number of elements in $\mathcal{T}_{j+1}$. Notice that 7.6.10 is a nonlinear equation and is often simplified in practive by replacing N_{j+1} by N_j.

5. Continue.

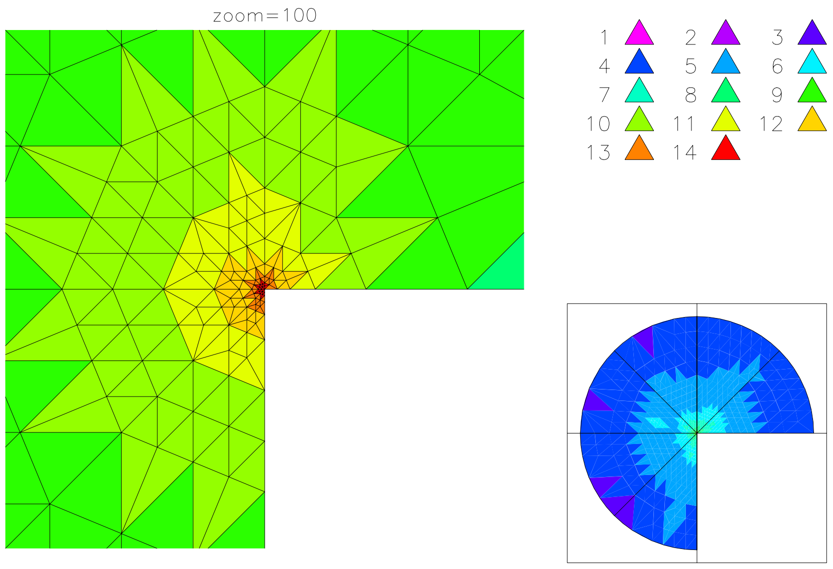


Figure 7.3: A typical adaptive grid generated by PLTMG

Chapter 8

Multigrid Methods for Structured Grids

In this chapter, we discuss multigrid methods for symmetric positive definite problems discretized on nested multilevel grids.

8.1 Analysis for smoothers

The most crucial step in developing a multigrid solver is the design of a relaxation scheme. A relaxation scheme is also the most problem-dependent part of a multigrid solver as most other parts (such as prolongation and restriction operators) are usually quite standard. The rôle of relaxation is not to reduce the overall error, but to smooth it out (namely damp out the non-smooth or high frequency components) so that it can be well approximated by functions on a coarser grid.

The smoother will be analysed by three approaches in this section. The first approach is through numerical experiments, which would give an intuitive idea of the numerical behavior of a smoother. The second approach is Brandt's local mode analysis. This approach, using local Fourier analysis, can give a good insight on the rôle of a smoother. The third approach is to build technical machinery for the convergence analysis of multigrid methods.

8.1-a A model problem and some numerical examples

Consider the Poisson equation with homogeneous Dirichlet condition on the unit square discretized with a uniform triangulation. The discretized equation can be expressed as

$$4u_{ij} - (u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}) = b_{i,j}, \quad 1 \le i, j \le n. \tag{8.1.1}$$

The damped Jacobi and Gauss–Seidel methods are among the most popular relaxation schemes for this problem. The damped Jacobi (or Richardson) iteration can be written as

$$4\tilde{u}_{ij} = \omega(\bar{u}_{i+1,j} + \bar{u}_{i-1,j} + \bar{u}_{i,j+1} + \bar{u}_{i,j-1}) + b_{i,j} \tag{8.1.2}$$

where $\tilde{u}_{ij}$ denotes the new value of u while $\bar{u}_{ij}$ denotes the old value of u, and the (point) Gauss–Seidel iteration (with lexicographical order on nodal points, from left to right and bottom to top):

$$4\tilde{u}_{ij} = (\bar{u}_{i+1,j} + \tilde{u}_{i-1,j} + \bar{u}_{i,j+1} + \tilde{u}_{i,j-1}) + b_{i,j}. \tag{8.1.3}$$

The Gauss–Seidel method has a good smoothing property. Let us illustrate this by a simple numerical example. Consider eqn. (8.1.1) with an initial residual $u - u^0$ shown on the left plot of Fig. **??**. The initial residual apparently contains a lot of oscillations. The middle plot in Fig. **??** is the residual after 2 Gauss–Seidel iterations. As we see the error components are smoothed out very quickly with only two Gauss–Seidel iterations although the global errors are still very large. The right plot in Fig. **??** is the residual after one hundred iterations and as we see the error is still quite big.

Basic ideas in a multigrid strategy The above numerical examples show that high frequency errors, which involve local variations in the solution, are well annihilated by simple relaxation methods such as Gauss–Seidel iterations. Low frequency or more global errors are much more insensitive to the application of simple relaxation methods. In fact, as shown in Fig. **??**, the convergence rate of the Gauss–Seidel iteration consists of a rather rapid initial residual reduction phase, which gradually develops into a much slower residual reduction phase, corresponding to a situation where all high frequency errors have been damped down and low frequency errors dominate. A multigrid methodology capitalizes on this rapid initial reduction of high frequency errors associated with an initial solution on the fine grid, using a simple relaxation scheme such as Gauss–Seidel iteration. Therefore, the solution is transferred to a coarse grid. On this grid, the low frequency errors of the fine grid manifest themselves as high frequency errors, and are thus damped out efficiently using the same relaxation scheme. The coarse grid corrections computed in this manner are interpolated back to the fine grid in order to update the solution. This procedure can be applied recursively on a sequence of coarser and coarser grids, where each grid-level is responsible for eliminating a particular frequency bandwidth of errors.

Multigrid strategies may be applied to any existing relaxation technique. The success of the overall solution strategy depends on a close matching between the bandwidth of errors which can be efficiently smoothed on a given grid using the particular chosen relaxation strategy, with a careful construction of a sequence of coarse grids, in order to represent the entire error frequency range.

8.1-b Brandt's local mode analysis

The local mode analysis of Brandt [?] is an effective general tool to analyze and predict the performance of a multigrid solver and in particular the performance of a smoother. This method is based on the fact that a relaxation process is often a local process in which the information propagates just a few mesh-sizes per sweep. Therefore, one can assume the problem to be in an unbounded domain, with constant (frozen) coefficients, in which case the algebraic error can be expanded in terms of Fourier series.

The local mode analysis for a smoother can in fact be applied in a rigorous fashion to the model problem discussed in the previous section. Let us first recall the discrete Fourier theory. For clarity, we confine our discussion to the two dimensional case. The discrete Fourier transform theory says that every discrete function $u : I_n \mapsto \mathbb{R}$, with $I_n = \{(i,j) : 0 \le i,j \le n\}$, can be written as

$$u_{i,j} = \sum_{\theta\in\Theta_n} c_\theta \psi_{i,j}(\theta), \quad \psi_{i,j}(\theta) = e^{\mathbf{i}(i\theta_1 + j\theta_2)}, \quad \mathbf{i} = \sqrt{-1}, \quad \theta = (\theta_1, \theta_2).$$

where

$$c_\theta = \frac{1}{(n+1)^2} \sum_{(k,l)\in I_n} u_{k,l}\psi_{k,l}(-\theta).$$

and

$$\Theta_n = \left\{ \frac{2\pi}{n+1}(k,l) \quad -m \le k,l \le m+p \right\},$$

where $p = 1, m = (n+1)/2$ for odd n and $p = 0, m = n/2 + 1$ for even n.

We now first use the discrete Fourier transform to analyze the damped Jacobi method. Let $\tilde{\epsilon}_{i,j} = u_{i,j} - \tilde{u}_{i,j}$ and $\bar{\epsilon}_{i,j} = u_{i,j} - \bar{u}_{i,j}$. It is easy to see that

$$\tilde{\epsilon}_{ij} = \bar{\epsilon}_{i,j} - \frac{\omega}{4}(4\bar{\epsilon}_{ij} - (\bar{\epsilon}_{i+1,j} + \bar{\epsilon}_{i-1,j} + \bar{\epsilon}_{i,j+1} + \bar{\epsilon}_{i,j-1})). \tag{8.1.4}$$

We write

$$\tilde{\epsilon}_{i,j} = \sum_{\theta\in\Theta_n} \tilde{c}_\theta \psi_{i,j}(\theta) \tag{8.1.5}$$

and

$$\bar{\epsilon}_{i,j} = \sum_{\theta\in\Theta_n} \bar{c}_\theta \psi_{i,j}(\theta). \tag{8.1.6}$$

Substituting the above expressions into (8.1.4) and comparing the coefficients of each $\psi_{ij}(\theta)$, we obtain that

$$\lambda(\theta) = 1 - \omega\left(1 - \frac{\cos\theta_1 + \cos\theta_2}{2}\right). \tag{8.1.7}$$

where

$$\lambda(\theta) \equiv \frac{\tilde{c}(\theta)}{\bar{c}(\theta)} \tag{8.1.8}$$

is called the amplification factor of the local mode $\psi_{i,j}(\theta)$.

The smoothing factor introduced by Brandt is the quantity

$$\bar{\rho} = \sup\{|\lambda(\theta)|, \pi/2 \le |\theta_k| \le \pi, k = 1, 2\}. \tag{8.1.9}$$

Roughly speaking, the smoothing factor $\bar{\rho}$ is the maximal amplification factor corresponding to those high frequency local modes that oscillate within range $2h$ (and hence cannot be resolved by a coarse grid of size $2h$).

For the damped Jacobi method, it is easy to see that

$$\bar{\rho} = \max\{|1 - 2\omega|, |1 - \omega/2|, |1 - 3\omega/2|\}.$$

The optimal ω that minimizes the smoothing factor is

$$\omega = 4/5, \quad \bar{\rho} = 3/5.$$

For $\omega = 1$ we have $\bar{\rho} = 1$. This means that the undamped Jacobi method for this model problem, although convergent as an iterative method by itself, should not be used as a smoother.

We next examine the smoothing property of the Gauss–Seidel iteration. Unlike the Jacobi method, the Gauss–Seidel method depends on the ordering of the unknowns. The most natural ordering is perhaps the lexicographic order which was used in the numerical examples given earlier and the corresponding Gauss–Seidel method reads

$$\tilde{\epsilon}_{ij} = \frac{1}{4}(\bar{\epsilon}_{i+1,j} + \tilde{\epsilon}_{i-1,j} + \bar{\epsilon}_{i,j+1} + \tilde{\epsilon}_{i,j-1}). \tag{8.1.10}$$

Again using the Fourier transform (8.1.5) and (8.1.6), we obtain the local amplification factor as follows:

$$\lambda(\theta) = \frac{e^{\mathbf{i}\theta_1} + e^{\mathbf{i}\theta_1}}{4 - e^{-\mathbf{i}\theta_1} - e^{-\mathbf{i}\theta_2}}.$$

It is elementary to see that

$$\bar{\rho} = |\lambda(\pi/2, \cos^{-1}(4/5))| = 1/2.$$

This means that the Gauss–Seidel method is a better smoother than the damped Jacobi method.

A more interesting ordering for the Gauss–Seidel method is the so-called red-black ordering. In this particular example, we say two grid points belong to the same color if and only they are not neighbors (in either the horizontal or vertical directions). It is easy to see that the uniform grid in our example can be grouped into two colors, often called red and black. The red-black ordering is to first order all the nodes in one color and then order the other points in another color. (The actual ordering within the same color is not crucial.)

The smoothing factor for the Gauss–Seidel method with red-black ordering cannot be obtained as easily as the lexicographical ordering, but it can indeed be proved that

$$\bar{\rho} = 1/4.$$

This means that the Gauss–Seidel method with red-black ordering is a better smoother than the one with the lexicographical ordering. Furthermore red-black Gauss–Seidel has much better parallel features.

8.1-c General smoother analysis

We shall now develop some technical results concerning the smoothing property of the Gauss–Seidel method. We choose to study the Gauss–Seidel method since it is one of the better smoothers for our model problems and also it is less obvious to analyse. The analysis for other smoothers is relatively simple (see the analysis for Richardson in (2.1.5)).

Lemma 8.1.11 *For the stiffness matrix $\mathcal{A} = \mathcal{D} - \mathcal{L} - \mathcal{U}$,*

$$\|(\mathcal{D} - \mathcal{L})\xi\|_2 \eqsim h^{d-2}\|\xi\|_2 \quad \forall\, \xi \in \mathbb{R}^N.$$

Proof. Because of the sparsity, it is trivial to prove that

$$\|(\mathcal{D} - \mathcal{L})\xi\|_2 \lesssim h^{d-2}\|\xi\|_2.$$

Now it follows that

$$\begin{aligned} h^{2-d}(\xi,\xi) &\lesssim \frac{1}{2}(\mathcal{D}\xi,\xi) \le \frac{1}{2}((\mathcal{A}+\mathcal{D})\xi,\xi) \\ &= ((\mathcal{D}-\mathcal{L})\xi,\xi) \le \|(\mathcal{D}-\mathcal{L})\xi\|_2\|\xi\|_2. \end{aligned}$$

This completes the proof.

The following result is an operator interpretation of the algebraic result given in Lemma 8.1.11, which means that Gauss–Seidel is basically like a Richardson iteration.

Lemma 8.1.12 *Assume that $R : \mathcal{V} \mapsto \mathcal{V}$ represents the iterator for symmetric Gauss–Seidel iteration. Then*

$$R \eqsim h^2 \eqsim \lambda_h^{-1}.$$

Proof. By definition, the matrix representation of R is

$$\tilde{R} = (\mathcal{D} - \mathcal{U})^{-1}\mathcal{D}(\mathcal{D} - \mathcal{L})^{-1}\mathcal{M}.$$

Given $v \in \mathcal{V}$, let $\nu = \tilde{v}$; then it is easy to see that

$$(Rv, v) = \|\mathcal{D}^{\frac{1}{2}}(\mathcal{D} - \mathcal{L})^{-1}\mathcal{M}\nu\|_2^2$$

Thus it is equivalent to showing that

$$\|\mathcal{D}^{\frac{1}{2}}(\mathcal{D} - \mathcal{L})^{-1}\mathcal{M}\nu\|_2^2 \eqsim h^2(\mathcal{M}\nu, \nu).$$

Making a change of variable $\xi = (\mathcal{D} - \mathcal{L})^{-1}\mathcal{M}\nu$ and using the fact that $\mathcal{M}^{-1} \eqsim h^{-d}$, the above relation can be reduced to

$$\|(\mathcal{D} - \mathcal{L})\xi\|_2^2 \eqsim h^{d-2}(\mathcal{D}\xi, \xi) \eqsim h^{2(d-2)}\|\xi\|_2^2,$$

which was given by Lemma 8.1.11

Lemma 8.1.13 *For the stiffness matrix $\mathcal{A} = \mathcal{D} - \mathcal{L} - \mathcal{U}$*

$$(\mathcal{A}\xi, \xi) \le \frac{2}{1 + k_0}((\mathcal{D} - \mathcal{L})\xi, \xi) \quad \forall\, \xi \in \mathbb{R}^N,$$

where k_0 is the maximal number of nonzero row entries of $\mathcal{A}$.

If $A, R : \mathcal{V} \mapsto \mathcal{V}$ represents the iterator for Gauss–Seidel iteration, then

$$(Av, v) \le \frac{2}{1 + k_0}(R^{-1}v, v) \quad \forall\, v \in V_h.$$

Proof. It is easy to see that the desired estimate is equivalent to the following:

$$(\mathcal{A}\xi, \xi) \le k_0(\mathcal{D}\xi, \xi) \quad \forall\, \xi \in \mathbb{R}^N$$

which can be obtained by a simple application of the Cauchy–Schwarz inequality.

8.2 A basic multigrid cycle: the backslash (\) cycle

Although, as we shall see, multigrid methods have many variants, there is one particular multigrid algorithm which can be viewed as a basic multigrid cycle. This algorithm is sometimes called the backslash (\) cycle (we shall explain below why this algorithm is given this name).

We shall first present this method from a more classical point of view. This more classical approach makes it easier to introduce many different kinds of classical multigrid methods and also make it possible to use more classical approaches to analyze the convergence of multigrid methods.

A multigrid process can be viewed as defining a sequence of operators $B_k : \mathcal{M}_k \mapsto \mathcal{M}_k$ which are approximate inverses of A_k in the sense that $\|I - B_kA_k\|_A$ is bounded away from 1. A typical way of defining such a sequence of operators is the following *backslash* cycle multigrid procedure.

Algorithm 8.2.14 *For $k = 0$, define $B_0 = A_0^{-1}$. Assume that $B_{k-1} : \mathcal{M}_{k-1} \mapsto \mathcal{M}_{k-1}$ is defined. We shall now define $B_k : \mathcal{M}_k \mapsto \mathcal{M}_k$ which is an iterator for the equation of the form*

$$A_k v = g.$$

1. Fine grid smoothing: *for $v^0 = 0$ and $l = 1, 2, \cdots, m$,*

$$v^l = v^{l-1} + R_k(g - A_k v^{l-1}).$$

2. Coarse grid correction: *$e_{k-1} \in \mathcal{M}_{k-1}$ is the approximate solution of the residual equation $A_{k-1}e = Q_{k-1}(g - Av^m)$ by the iterator B_{k-1}:*

$$e_{k-1} = B_{k-1}Q_{k-1}(g - Av^m).$$

Define

$$B_k g = v^m + e_{k-1}.$$

After the first step, the residual $v - v^m$ is small at high frequencies. In other words, $v - v^m$ is smoother (see the middle plot in Fig. **??**) and hence it can be very well approximated by a coarse space $\mathcal{M}_{k-1}$. The second step in the above algorithm plays the rôle of correcting the low frequencies by the coarser space $\mathcal{M}_{k-1}$ and the coarse grid solver B_{k-1} given by induction.

With the above defined B_k, we may consider the following simple iteration:

$$u^{k+1} = u^k + B_J(f - Au^k). \tag{8.2.15}$$

There are many different ways to make use of B_k, which will be discussed later.

Before we study its convergence, we now discuss briefly the algebraic version of the above algorithm.

Let $\Phi^k = (\phi_1^k, \cdots, \phi_{n_k}^k)$ be the nodal basis vector for the space $\mathcal{M}_k$; we define the so-called prolongation matrix $\mathcal{I}_k^{k+1} \in \mathbb{R}^{n_{k+1} \times n_k}$ as follows

$$\Phi^k = \Phi^{k+1}\mathcal{I}_k^{k+1}. \tag{8.2.16}$$

Algorithm 8.2.17 (Matrix version) *Let $\mathcal{B}_0 = \mathcal{A}_0^{-1}$. Assume that $\mathcal{B}_{k-1} \in \mathbb{R}^{n_{k-1} \times n_{k-1}}$ is defined; then for $\eta \in \mathbb{R}^{n_k}$, $\mathcal{B}_k \in \mathbb{R}^{n_k \times n_k}$ is defined as follows.*

1. Fine grid smoothing: *for $\nu^0 = 0$ and $l = 1, 2, \cdots, m$*

$$\nu^l = \nu^{l-1} + \mathcal{R}_k(\eta - \mathcal{A}_k \nu^{l-1}).$$

2. Coarse grid correction: *$\varepsilon_{k-1} \in \mathbb{R}^{n_{k-1}}$ is the approximate solution of the residual equation $\mathcal{A}_{k-1}\varepsilon = (\mathcal{I}_{k-1}^k)^t(\eta - \mathcal{A}_k\nu^m)$ by using $\mathcal{B}_{k-1}$,*

$$\varepsilon_{k-1} = \mathcal{B}_{k-1}(\mathcal{I}_{k-1}^k)^t(\eta - \mathcal{A}_k\nu^m).$$

Define

$$\mathcal{B}_k\eta = \nu^m + \mathcal{I}_{k-1}^k\varepsilon_{k-1}.$$

The above algorithm is given in recurrence form, but it can also be easily implemented in a nonrecursive fashion. For such an implementation, we refer to Algorithm 8.4.24

8.3 A convergence analysis using full elliptic regularity

With the assumption of full elliptic regularity (for instance, the domain is Lipschitzan, see also Theorem (7.3.11)), a very sharp convergence estimate can be obtained in a very simple and elegant fashion.

We shall assume that the smoothers R_k are SPD and satisfy

$$(8.3.18) \qquad \frac{c_0}{\lambda_k}(v,v) \le (R_k v, v) \le (A_k^{-1} v, v) \quad \forall\, v \in \mathcal{M}_k.$$

We would like to remark that the above assumptions on R_k can be much weakened and, for example, the R_k do not need to be symmetric (in this case, assumptions need to be made on the symmetrizations of the R_k, see, e.g., §8.5).

If Ω is a bounded Lipschtiz domain, then there exists a positive constant c_1 independent of mesh parameters such that (see Theorem (7.3.11))

$$(8.3.19) \qquad \|(I-P_{k-1})v\|_A^2 \le c_1 \lambda_k^{-1} \|A_k v\|^2 \quad \forall\, v \in \mathcal{M}_k.$$

The next technical result shows that any function smoothened by local relaxation can be well approximated by a coarser grid.

Lemma 8.3.20

$$\|(I-P_{k-1})K_k^m v\|_A^2 \le \frac{c_1}{2mc_0}(\|v\|_A^2 - \|K_k^m v\|_A^2).$$

Proof.

$$\begin{aligned} \|(I-P_{k-1})K_k^m v\|_A^2 &\le c_1 \lambda_k^{-1} \|A_k K_k^m v\|^2 \\ &= \frac{c_1}{c_0}(R_k A_k K_k^m v, A_k K_k^m v) = \frac{c_1}{c_0}((I-K_k)K_k^{2m} v, v)_A \\ &\le \frac{c_1}{2mc_0}(\|v\|_A^2 - \|K_k^m v\|_A^2). \end{aligned}$$

The proof is completed by using the following elementary inequality:

$$(8.3.21) \qquad \begin{aligned} ((I-K_k)K_k^{2m} v, v)_A &\le \frac{1}{2m}\sum_{j=0}^{2m-1}((I-K_k)K_k^j v, v)_A \\ &\le \frac{1}{2m}(\|v\|_A^2 - \|K_k^m v\|_A^2). \end{aligned}$$

Theorem 8.3.22 *For the Algorithm 8.2.14, we have*

$$\|I - B_k A_k\|_A^2 \le \frac{c_1}{2mc_0 + c_1}, \quad 1 \le k \le J.$$

Proof. By definition of Algorithm 8.2.14, we have

$$I - B_k A_k = (I - P_{k-1} B_{k-1} A_{k-1})(I - R_k A_k)^m$$

and, thus, for all $v \in \mathcal{M}_k$

$$\begin{aligned} & \|(I - B_k A_k)v\|_A^2 \\ = \; & \|(I - P_{k-1})K_k^m v\|_A^2 + \|(I - B_{k-1}A_{k-1})P_{k-1}K_k^m v)\|_A^2. \end{aligned}$$

Let $\delta = c_1/(2mc_0 + c_1)$. We shall prove the above estimate by induction. First of all it is obviously true for $k = 0$. Assume it holds for $k - 1$. In the case of k, we have from the above identity that

$$\begin{aligned} \|(I - B_k A_k)v\|_A^2 & \le \|(I - P_{k-1})K_k^m v\|_A^2 + \delta \|P_{k-1}K_k^m v\|_A^2 \\ & \le (1 - \delta)\|(I - P_{k-1})K_k^m v\|_A^2 + \delta \|K_k^m v\|_A^2 \\ & \le (1 - \delta)\frac{c_1}{2mc_0}(\|v\|_A^2 - \|K_k^m v\|_A^2) + \delta \|K_k^m v\|_A^2 \\ & \le \delta \|v\|_A^2. \end{aligned}$$

8.4 V-cycle and W-cycle

Two important variants of the above backslash cycle are the so-called V-cycle and W-cycle.

A V-cycle algorithm is obtained from the backslash cycle by performing more smoothings after the coarse grid corrections. Such an algorithm, roughly speaking, is like a backslash (\) cycle plus a slash (/) (a reversed backslash) cycle. The detailed algorithm is given as follows.

Algorithm 8.4.23 *For $k = 0$, define $B_0 = A_0^{-1}$. Assume that $B_{k-1} : \mathcal{M}_{k-1} \mapsto \mathcal{M}_{k-1}$ is defined. We shall now define $B_k : \mathcal{M}_k \mapsto \mathcal{M}_k$ which is an iterator for the equation of the form*

$$A_k v = g.$$

1. Pre-smoothing: *for $v^0 = 0$ and $l = 1, 2, \cdots, m$*

$$v^l = v^{l-1} + R_k(g - A_k v^{l-1}).$$

2. Coarse grid correction: *$e_{k-1} \in \mathcal{M}_{k-1}$ is the approximate solution of the residual equation $A_{k-1}e = Q_{k-1}(g - Av^m)$ by the iterator B_{k-1}:*

$$e_{k-1} = B_{k-1}Q_{k-1}(g - Av^m).$$

3. Post-smoothing: *for $v^{m+1} = v^m + e_{k-1}$ and $l = m + 2, 2, \cdots, 2m$*

$$v^l = v^{l-1} + R_k(g - A_k v^{l-1}).$$

A nonrecursive implementation The recursive formulation of the above algorithm makes it a little less straightforward to code sometimes. A nonrecursive version of the algorithm is given below in terms of matrices and vectors.

Algorithm 8.4.24 (V-cycle computation of $\mathcal{B}\beta$)

```
$\beta_J = \beta$,
for $l = J : 1$,                                   for $l = 2 : J$,
    $\alpha_l = \mathcal{R}_l \beta_l$,                          $\tilde{\alpha}_l = \alpha_l + \mathcal{I}_{l-1}^l \alpha_{l-1}$;
    $\beta_{l-1} = (\mathcal{I}_{l-1}^l)^t (\beta_l - \mathcal{A}_l \alpha_l)$;     $\alpha_l = \tilde{\alpha}_l \mathcal{R}_l^t (\beta_l - \mathcal{A}_l \tilde{\alpha}_l)$;
endfor                                             endfor
    $\mathcal{B}\beta = \alpha_J$.
```

The reason why this algorithm is called a V-cycle is quite clear with the above implementation. The algorithm starts on the finest level and traverses all the grids, one at a time, until it reaches the coarsest grid. Then it traverses all the grids until it reaches the finest level.

It is easy to see that the operators B_k defined by the above V-cycle algorithm satisfy

$$I - B_k A_k = (I - B_k^{(\backslash)} A_k)^* (I - B_k^{(\backslash)} A_k)$$

where the $B_k^{(\backslash)}$ correspond to the operators defined by the backslash cycle (Algorithm 8.2.14) and $*$ is the adjoint operator with respect to the A-inner product. Consequently,

$$\|I - B_k A_k\|_A = \|I - B_k^{(\backslash)} A_k\|_A^2.$$

This means that the convergence of the V-cycle is a consequence of the convergence of the backslash cycle.

The W-cycle, roughly speaking, is like a V-cycle plus another V-cycle. In a V-cycle iteration, the coarse grid correction is only performed once, while in a W-cycle, the coarse grid correction is performed twice.

Algorithm 8.4.25 *For $k = 0$, define $B_0 = A_0^{-1}$. Assume that $B_{k-1} : \mathcal{M}_{k-1} \mapsto \mathcal{M}_{k-1}$ is defined. We shall now define $B_k : \mathcal{M}_k \mapsto \mathcal{M}_k$ which is an iterator for the equation of the form*

$$A_k v = g.$$

1. Pre-smoothing: *for $v^0 = 0$ and $l = 1, 2, \cdots, m$*

$$v^l = v^{l-1} + R_k(g - A_k v^{l-1}).$$

2. Coarse grid correction: *$e_{k-1} \in \mathcal{M}_{k-1}$ is the approximate solution of the residual equation $A_{k-1} e = Q_{k-1}(g - A v^m)$ by applying iterator B_{k-1} twice, $e_{k-1} = w^2$, with $w^0 = 0$ and*

$$w^j = w^{j-1} + B_{k-1}(Q_{k-1}(g - A v^m) - A_{k-1} w^{j-1}), \quad j = 0, 1, 2.$$

3. Post-smoothing: *for $v^{m+1} = v^m + e_{k-1}$ and $l = m + 2, 2, \cdots, 2m$*

$$v^l = v^{l-1} + R_k(g - A_k v^{l-1}).$$

8.5 Subspace correction interpretation

We shall now discuss the multigrid method from the subspace correction point of view. Let $\mathcal{M}_k$ $(k = 0, 1, \cdots, J)$ be the multilevel finite element spaces defined as in the preceding section. Again let $\mathcal{V} = \mathcal{M}_J$, but set $\mathcal{V}_k = \mathcal{M}_{J-k}$. In this case, the decomposition (5.1.1) is trivial.

We observe that, with the above choice of subspaces $\mathcal{V}_k$, there are redundant overlappings in the decomposition (5.1.1). The point is that these overlappings can be used advantageously in the choice of the subspace solvers in a simple fashion. Roughly speaking, the subspace solvers need only take care of those 'nonoverlapped parts' (which correspond to the so-called *high frequencies*). As we know, methods like the Gauss–Seidel method discussed earlier just satisfy such requirements.

With the above ingredients, the successive subspace correction method Algorithm 5.2.13 can be stated as follows.

Algorithm 8.5.26 *Given* $u^0 \in \mathcal{V}$.

```
for $k = 0, 1, \ldots$ till convergence
  $v \leftarrow u^k$
  for $i = J : -1 : 0$   $v \leftarrow v + R_i Q_i (f - Av)$   endfor
  $u^{k+1} \leftarrow v$.
endfor
```

Lemma 8.5.27 *For the Algorithm 8.2.14 with* $m = 1$*, we have*

$$I - B_J A_J = (I - T_0)(I - T_1) \cdots (I - T_J) \tag{8.5.28}$$

where

$$T_0 = P_0, T_k = R_k A_k P_k, \quad 1 \le k \le J.$$

Hence, with such defined operators B_k*, the iteration* (8.2.15) *is mathematically equivalent to Algorithm 8.5.26.*

Furthermore if B_J^m *is obtained from Algorithm 8.2.14 for general* $m \ge 1$*, then*

$$\|I - B_J^m A_J\|_A \le \|I - B_J A_J\|_A.$$

Based on the above lemma, different proofs may be obtained of the convergence of the backslash cycle multigrid method.

Theorem 8.5.29 *For the iteration* (8.2.15) *with* B_J *given by Algorithm 8.2.14 with one smoothing on each level, then*

$$\|I - B_J A_J\|_A^2 \le 1 - \frac{c_0}{c_1}.$$

Proof. Denote $E_{-1} = I$ and for $0 \le i \le J$,

$$E_i = (I - T_i)(I - T_{i-1}) \cdots (I - T_1)(I - T_0).$$

Note that $E_J = (I - B_J A_J)^*$. It follows that, denoting $\tilde{P}_i = P_i - P_{i-1}$,

$$\begin{aligned}
\|v\|_A^2 &= \sum_{i=0}^{J} (\tilde{P}_i v, v)_A \\
&= \sum_{i=0}^{J} (\tilde{P}_i v, E_{i-1} v)_A \quad (\text{since } (I - E_{i-1})v \in \mathcal{M}_{i-1}) \\
&= \sum_{i=0}^{J} (\tilde{P}_i v, A_i P_i E_{i-1} v) \\
&\le \sqrt{c_1} \sum_{i=0}^{J} \lambda_i^{-1/2} \|\tilde{P}_i v\|_A \|A_i P_i E_{i-1} v\| \quad (\text{by } (8.3.19)) \\
&\le \sqrt{\frac{c_1}{c_0}} \sum_{i=0}^{J} \|\tilde{P}_i v\|_A (R_i A_i P_i E_{i-1} v, A_i P_i E_{i-1} v)^{1/2} \quad (\text{by } (8.3.18)) \\
&\le \sqrt{\frac{c_1}{c_0}} \sum_{i=0}^{J} \|\tilde{P}_i v\|_A (T_i E_{i-1} v, E_{i-1} v)_A^{1/2} \\
&\le \sqrt{\frac{c_1}{c_0}} \left(\sum_{i=0}^{J} \|\tilde{P}_i v\|_A^2 \right)^{1/2} \left(\sum_{i=0}^{J} (T_i E_{i-1} v, E_{i-1} v)_A \right)^{1/2} \\
&\le \sqrt{\frac{c_1}{c_0}} \left(\sum_{i=0}^{J} \|\tilde{P}_i v\|_A^2 \right)^{1/2} \left(\sum_{i=0}^{J} (T_i E_{i-1} v, E_{i-1} v)_A \right)^{1/2} \\
&\le \sqrt{\frac{c_1}{c_0}} \|v\|_A \left(\|v\|_A^2 - \|E_J v\|_A^2 \right)^{1/2}.
\end{aligned}$$

Consequently,

$$\|E_J v\|_A^2 \le \left(1 - \frac{c_0}{c_1} \right) \|v\|_A^2 \quad \forall\, v \in \mathcal{V}.$$

The desired estimate then follows easily.

8.6 Full multigrid cycle

Use multigrid iteration to get the finite element solution within the truncation error, we need $O(\log n_J)$ MG iterations but each iteration costs $O(n_J)$ operations. Thus, the total computational complexity is $O(n_J \log n_J)$ (not quite optimal).

We shall now describe a more efficient multigrid technique, called a full multigrid cycle, originally proposed by Brandt.

On each level of the finite element space $\mathcal{V}_k$, there is a corresponding finite element approximation $u^{(k)} \in \mathcal{V}_k$ such that

$$A(u^{(k)}, v) = (F, v) \quad \forall v \in \mathcal{V}. \tag{8.6.30}$$

Similarly to (7.3.11), the best error estimate in the H^1 norm is

$$\|U - u^{(k)}\|_1 = O(h_k^\alpha), \tag{8.6.31}$$

if for any $F \in H^{\alpha-1}(\Omega)$, there exists $U \in H_0^1(\Omega) \cap H^{1+\alpha}(\Omega)$ such that

$$A(U, v) = (F, v), \quad \forall v \in H_0^1(\Omega).$$

Here $\alpha \in (0, 1]$. If $\mu^{(k)} \in \mathbb{R}^{n_k}$ is the nodal value vector of $u^{(k)}$, then

$$\mathcal{A}_k \mu^{(k)} = b^{(k)} \tag{8.6.32}$$

where $b^{(k)} = ((F, \phi_i^k))$. It can be proved that, with $\mathcal{I}_k^{k+1}$ given by (8.2.16),

$$b^{(k)} = (\mathcal{I}_k^{k+1})^t b^{(k+1)} \quad \text{with} \quad b^{(J)} = b.$$

The full multigrid method is based on the following two observations.

1. $u^{(k-1)} \in \mathcal{V}_{k-1} \subset \mathcal{V}_k$ is close to $u^{(k)} \in \mathcal{V}_k$ and hence can be used as an initial guess for an iterative scheme for solving $u^{(k)}$.
2. Each $u^{(k)}$ can be solved within its truncation error shown in (8.6.31) by a multigrid iterative scheme.

Algorithm 8.6.33 $\hat{\mu}^{(1)} \leftarrow \mathcal{A}_1^{-1} b^{(1)}$. *For* $k = 2 : J$

1. $\hat{\mu}^{(k)} \leftarrow \mathcal{I}_{k-1}^k \hat{\mu}^{(k-1)}$,

2. iterate $\hat{\mu}^{(k)} \leftarrow \hat{\mu}^{(k)} + \mathcal{B}_k(b^{(k)} - \mathcal{A}_k \hat{\mu}^{(k)})$ *m times.*

The most important fact about the full multigrid method is that it has an optimal computational complexity $O(N)$ to compute the solution within truncation error.

Proposition 8.6.34 *Assume that that C_0 is a positive constant satisfying (for all k)*

$$\|\mu^{(k)} - \mathcal{I}_{k-1}^k \mu^{(k-1)}\|_{\mathcal{A}} \le C_0 h_k^\alpha.$$

Then

$$\|\mu^{(k)} - \hat{\mu}^{(k)}\|_{\mathcal{A}} \le h_k^\alpha$$

if

$$m \ge \frac{\log(2^\alpha + C_0)}{|\log \delta|}.$$

Proof. By definition, $\|\mu^{(1)} - \hat{\mu}^{(1)}\|_{\mathcal{A}} = 0$. Now assume that

$$\|\mu^{(k-1)} - \hat{\mu}^{(k-1)}\|_{\mathcal{A}} \le h_{k-1}^\alpha.$$

Then

$$\begin{aligned}
\|\mu^{(k)} - \hat{\mu}^{(k)}\|_{\mathcal{A}} &\le \delta^m \|\mu^{(k)} - \mathcal{I}_{k-1}^k \hat{\mu}^{(k-1)}\|_{\mathcal{A}} \\
&\le \delta^m \|\mu^{(k)} - \mathcal{I}_{k-1}^k \mu^{(k-1)}\|_{\mathcal{A}} + \delta^m \|\mu^{(k-1)} - \hat{\mu}^{(k-1)}\|_{\mathcal{A}} \\
&\le \delta^m (C_0 + 2^\alpha) h_k^\alpha \\
&\le h_k^\alpha
\end{aligned}$$

8.7 BPX multigrid preconditioners

We shall now describe a parallelized version of the multigrid method studied earlier. This method was first proposed by Bramble, Pasciak and Xu [?] and by Xu [?], and is now often known as the BPX preconditioner in the literature.

8.7-a Basic algorithm and theory

There are different ways of deriving the BPX preconditioners. The method originally resulted from an attempt to parallelize the classical multigrid method. With the current multigrid theoretical technology, the derivation of this method is not so difficult. We shall first derive this preconditioner based on Theorem **??** and then, in the next subsection, study the method using the framework of subspace correction.

One can prove that

$$(Av, v) \eqsim \sum_{k=0}^{J} h_k^{-2} \|(Q_k - Q_{k-1})v\|^2 = (\hat{A}v, v), \quad v \in \mathcal{V},$$

where

$$\hat{A} = \sum_k h_k^{-2}(Q_k - Q_{k-1}).$$

Note that

$$\hat{A}^{-1} = \sum_k h_k^{2}(Q_k - Q_{k-1}).$$

Using the fact that $h_k = 2h_{k+1}$, we deduce that

$$\begin{aligned}
(\hat{A}^{-1}v, v) &= \sum_{k=0}^{J} h_k^2((Q_k - Q_{k-1})v, v) \\
&= \sum_{k=0}^{J} h_k^2(Q_k v, v) - \sum_{k=0}^{J-1} h_{k+1}^2(Q_k v, v) \\
&\eqsim h_J^2(v, v) + \sum_{k=0}^{J-1} h_k^2(Q_k v, v) \\
&\eqsim \sum_{k=0}^{J} h_k^2(Q_k v, v) = (\tilde{B}v, v)
\end{aligned}$$

where

$$\tilde{B} = \sum_{k=0}^{J} h_k^2 Q_k.$$

If $R_k : \mathcal{M}_k \mapsto \mathcal{M}_k$ is an SPD operator satisfying

$$(R_k v_k, v_k) \eqsim h_k^2(v_k, v_k) \quad \forall\, v_k \in \mathcal{M}_k \tag{8.7.35}$$

then, for

$$B = \sum_{k=0}^{J} R_k Q_k \tag{8.7.36}$$

we have

$$(Bv, v) \eqsim (\tilde{B}v, v) \eqsim (A^{-1}v, v),$$

namely

$$\kappa(BA) \eqsim 1.$$

Theorem 8.7.37 *Assume that the R_k satisfy* (8.7.35)*; then the preconditioner* (5.2.7) *satisfies*

$$\kappa(BA) \eqsim 1.$$

We note that all the relaxation methods mentioned earlier, such as Richardson, Jacobi and symmetric Gauss–Seidel, satisfy (8.7.35).

Subspace correction approach

In §8.5, the slash cycle multigrid method is interpreted as a successive subspace correction method. Correspondingly, the BPX preconditioner can be interpreted as the relevant PSC (parallel subspace correction) preconditioner. It is possible to use the abstract theory in §5 to derive some estimates for the BPX preconditioner somewhat more refined than that in Theorem 8.7.37; we leave the details to the interested readers. In §8.9, we shall use this approach to analyze the BPX preconditioner for locally refined meshes.

Implementation Again, in view of (8.7.36), the algebraic representation of the preconditioner given by (8.7.36) is

$$\mathcal{B} = \sum_{k=0}^{J} \mathcal{I}_k \mathcal{R}_k \mathcal{I}_k^t, \tag{8.7.38}$$

where $\mathcal{I}_k \in \mathbb{R}^{n \times n_k}$ is the representation matrix of the nodal basis $\{\phi_i^k\}$ in $\mathcal{M}_k$ in terms of the nodal basis $\{\phi_i\}$ of $\mathcal{M}$, i.e.

$$(\phi_1^k, \cdots, \phi_{n_k}^k) = (\phi_1, \cdots, \phi_n)\mathcal{I}_k.$$

Let $\mathcal{I}_k^{k+1} \in \mathbb{R}^{n_{k+1} \times n_k}$ be as defined in (8.2.16); then

$$\mathcal{I}_k = \mathcal{I}_{J-1}^J \cdots \mathcal{I}_{k+1}^{k+2} \mathcal{I}_k^{k+1}.$$

This identity is very useful for the efficient implementation of

If $\mathcal{R}_k$ are given by the Richardson iteration, we have

$$\mathcal{B} = \sum_{k=0}^{J} h_k^{2-d} \mathcal{I}_k \mathcal{I}_k^t. \tag{8.7.39}$$

From (8.7.38) or (8.7.39), we see that the preconditioner depends entirely on the transformation between the nodal bases on multilevel spaces.

For $1 \le l \le J$, let

$$\mathcal{B}_l = \sum_{k=0}^{l} \mathcal{I}_k^l \mathcal{R}_k (\mathcal{I}_k^l)^t.$$

By definition $\mathcal{B} = \mathcal{B}_J$ and

$$\mathcal{B}_l = \mathcal{R}_l + \mathcal{I}_{l-1}^l \mathcal{B}_{l-1} (\mathcal{I}_{l-1}^l)^t.$$

We shall use the above recurrence relation to compute the action of $\mathcal{B}$. Assume that m_l is the number of operations that are needed to compute the action $\mathcal{B}_l \alpha_l$ for $\alpha_l \in \mathbb{R}^{n_l}$. By the identity

$$\mathcal{B}_l \alpha_l = \mathcal{R}_l \alpha_l + \mathcal{I}_{l-1}^l [B_{l-1} (\mathcal{I}_{l-1}^l)^t \alpha_l]$$

we get, for some constant $c_0 > 0$,

$$m_l \le m_{l-1} + c_0 n_l$$

from which we conclude that

$$m_J \le m_1 + c_0 \sum_{l=2}^{J} n_l \le c_1 n$$

for some positive constant c_1. This means that the action of B_J can be carried out within $O(n)$ operations.

Algorithm 8.7.40 (Computation of $\mathcal{B}\alpha$)

```
$\alpha_J = \alpha$;
for $l = J : 1$,
    $\alpha_{l-1} = (\mathcal{I}_{l-1}^l)^t \alpha_l$;
end

$\beta_0 = \mathcal{R}_0 \alpha_0$;
for $l = 1 : J$,
    $\beta_l = \mathcal{R}_l \alpha_l + \mathcal{I}_{l-1}^l \beta_{l-1}$;
end
$\mathcal{B}\alpha = \beta_J$.
```

As discussed above, the number of operations needed in the above algorithm is $O(n)$. We also note that all the vectors α_l for $1 \le k \le J$ need to be stored, but the whole storage space for these vectors is also only $O(n)$.

8.8 Hierarchical basis methods

Assume that we are given a nested sequence of multigrid subspaces of $H_0^1(\Omega)$,

$$\mathcal{M}_1 \subset \mathcal{M}_2 \subset \ldots \subset \mathcal{M}_k \subset \ldots \subset \mathcal{M}_J,$$

as described in Chapter 6. The so-called *hierarchical basis* refers to the special set of nodal basis functions

$$\{\phi_i^k : x_i^k \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}, k = 0, \cdots, J\}. \tag{8.8.41}$$

It is easy to see that this set of functions does form a basis of $\mathcal{M}$. For $d \neq 2$, it is often more convenient to use the scaled HB (hierarchical basis) as follows

$$\{h_k^{2-d}\phi_i^k : x_i^k \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}, k = 0, \cdots, J\}. \tag{8.8.42}$$

With a proper ordering, we shall denote the scaled HB by $\{\psi_i, i = 1 : N\}$.

The HB in multiple dimensions is formally a direct generalization of the HB in the one dimensional case. But the property for the corresponding stiffness matrix in multiple dimensions is not at all as clear as in one dimension where the stiffness matrix is an identity matrix in some special cases. In this section, we shall show that at least in two dimensions, a hierarchical basis is still very useful.

The hierarchical basis in two dimensions was first analysed by Yserentant in his pioneering paper [?]. The work of Yserentant was apparently motivated by the famous unpublished technical report of Bank and Dupont [?]. Incidently these three authors got together and wrote another important paper [?] on a Gauss–Seidel (or multiplicative) variant of the hierarchical basis method. The presentation of the materials in this section is of course mostly based on the aforementioned papers, and moreover it also adopts the view of subspace correction from Xu [?] (see also Xu [?]).

8.8-a Preliminaries

We shall now discuss multigrid subspaces that are directly related to the HB. Consider the part of the HB functions on level k as follows

$$\{\phi_i^k : x_i^k \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}\}. \tag{8.8.43}$$

It is easy to see that the above set of functions spans the subspace

$$\mathcal{V}_k = (I_k - I_{k-1})\mathcal{M} = (I - I_{k-1})\mathcal{M}_k, \quad \text{for } k = 0 : J. \tag{8.8.44}$$

Here, we recall, $I_{-1} = 0$ and $I_k : \mathcal{M} \mapsto \mathcal{M}_k$ is the nodal value interpolant. The above subspaces obviously give rise to a direct sum decomposition of the space $\mathcal{M}$ as follows:

$$\mathcal{V} = \oplus_{k=0}^J \mathcal{V}_k.$$

In fact, for any $v \in \mathcal{V}$, we have the following unique decomposition:

$$v = \sum_{k=0}^J v_k \quad \text{with} \quad v_k = (I_k - I_{k-1})v.$$

With the subspaces $\mathcal{V}_k$ given by (8.8.44), the operators A_k are all well conditioned. In fact, by

$$\|v\|_1 \lesssim h^{-1}\|v\|_0, \quad \mathcal{V} \in \mathcal{V}$$

and

$$\|(I_k - I_{k-1})v\|^2 + h_k^2\|I_k v\|_1^2 \lesssim c_d(k)h_k^2\|v\|_1^2, \quad \forall v \in \mathcal{V},$$

we can see that

$$A(v,v) \eqsim h_k^{-2}\|v\| \quad \forall\, v \in \mathcal{V}_k.$$

Here $c_d(k) = 1, J-k$ and $2^{(d-2)(J-k)}$ for $d = 1,2$ and $d \geq 3$, respectively As a result, the subspace equations can be solved effectively by elementary iterative methods such as Richardson, Jacobi and Gauss–Seidel methods.

8.8-b Stiffness matrix in terms of hierarchical basis

The easiest way of understanding the HB is perhaps, like in one dimension, through the study of the property of the corresponding stiffness matrix. As one may expect, the condition number of the HB stiffness matrix should be smaller than the NB (normal basis) stiffness matrix. This is indeed the case in two and three dimensions.

Theorem 8.8.45 *Assume that $\hat{\mathcal{A}}$ is the stiffness matrix under the scaled hierarchical basis; then*

$$\kappa(\hat{\mathcal{A}}) \lesssim \kappa_d(h) \tag{8.8.46}$$

where

$$\kappa_d(h) \lesssim \begin{cases} 1 & \text{if } d=1; \\ |\log h|^2 & \text{if } d=2; \\ h^{2-d} & \text{if } d \geq 3. \end{cases} \tag{8.8.47}$$

In fact, the estimates given in the above theorem can be proven to be sharp. The most interesting case is obviously $d = 2$ for which $\kappa(\hat{\mathcal{A}}) \lesssim |\log h|^2$. Compared with the conditioning of the stiffness matrix under the NB, this is a great improvement. It is also in the case $d = 2$ that the HB is most useful. Indeed for $d = 3$, the $\kappa(\hat{\mathcal{A}}) = O(h^{-1})$ is also one magnitude smaller than the condition number of the NB stiffness matrix, but such an improvement is not attractive as we shall see that a much better approach (such as the BPX preconditioner) is available. There is no doubt that, as far as $\kappa(\hat{\mathcal{A}})$ is concerned, the HB is of no use for $d \geq 4$.

The proof ca be found in Yserentant [?] (and see also Ong [?] for $d = 3$).

8.8-c Subspace correction approach and general case

Following Xu [?], we shall now study the HB method from the viewpoint of space decomposition and subspace correction.

In view of (5.2.7), the algebraic representation of the PSC preconditioner is

$$\mathcal{H} = \sum_{k=0}^{J} \mathcal{S}_k \mathcal{R}_k \mathcal{S}_k^t, \tag{8.8.48}$$

where $\mathcal{S}_k \in \mathbb{R}^{n\times(n_k-n_{k-1})}$ is the representation matrix of the nodal basis $\{\phi_i^k\}$ in $\mathcal{M}_k$, with $x_i^k \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}$, in terms of the nodal basis $\{\phi^i\}$ of $\mathcal{M}$.

A special case If $\mathcal{R}_k$ is given by the Richardson iteration $\mathcal{R}_k = h_k^{2-d}\mathcal{I}$, we have

$$\mathcal{H} = \sum_{k=0}^{J} h_k^{2-d}\mathcal{S}_k\mathcal{S}_k^t = \hat{S}\hat{S}^t.$$

where

$$\mathcal{S} = (h_1^{1-d/2}\mathcal{S}_1, h_2^{1-d/2}\mathcal{S}_2, \cdots, h_J^{1-d/2}\mathcal{S}_J)$$

is the representation matrix of the HB in terms of NB. Obviously the HB stiffness matrix and NB stiffness matrix are related by $\hat{\mathcal{A}} = \mathcal{S}^t\mathcal{A}\mathcal{S}$. Therefore,

$$\kappa(\mathcal{H}\mathcal{A}) = \kappa(\hat{\mathcal{A}}),$$

and, as a result of Theorem 8.8.45,

$$\kappa(\mathcal{H}\mathcal{A}) \eqsim \kappa_d(h). \tag{8.8.49}$$

The above estimate apparently also holds for the more general $\mathcal{H}$ when $\mathcal{R}_k$ is given by either Jacobi or symmetric Gauss–Seidel since either of these iterations satisfies the following spectral equivalence:

$$\mathcal{R}_k \eqsim h_k^{2-d}. \tag{8.8.50}$$

In fact, the estimate (8.8.49) also follows easily from the general theory for the PSC preconditioner.

For the SSC iterative method, it is more convenient to choose $\mathcal{R}_k$ to be the symmetric Gauss–Seidel as the other two methods need to be properly scaled to assume that $\omega_1 \in (0, 2)$. The resulting algorithm is as follows.

Algorithm 8.8.51 *Let $\mu^0 \in \mathbb{R}^n$ be given. Assume that $\mu^k \in \mathbb{R}^n$ is obtained. Then μ^{l+1} is defined by*

$$\mu^{l+(k+1)/(J+1)} = \mu^{l+k/(J+1)} + \mathcal{S}_k\mathcal{R}_k\mathcal{S}_k^t(\eta - \mathcal{A}\mu^{l+(i-1)/J})$$

for $k = 0 : J$.

It can be proved later that

$$\|\mu - \mu_l\|_{\mathcal{A}} \le \left(1 - \frac{c}{\kappa_d(h)}\right)^l \|\mu - \mu_0\|_{\mathcal{A}}. \tag{8.8.52}$$

8.8-d Convergence analysis

Lemma 8.8.53 *We assume that R_k is either Richardson, or Jacobi or symmetric Gauss–Seidel iteration; then*

$$\lambda_k^{-2}\|v\|_A^2 \lesssim (R_k A_k v, v)_A \le \omega_1 (v,v)_A, \quad \forall\, v \in \mathcal{V}_k.$$

where ω_1 is a constant and for symmetric Gauss–Seidel, $\omega_1 = 1$.

Proof. The proof for the Richardson or Jacobi method is straightforward. The proof for the symmetric Gauss–Seidel method is almost identical to that of Lemma 8.1.12 and the detail is left to the reader.

Lemma 8.8.54

$$K_0 \lesssim c_d \quad \textit{and} \quad K_1 \lesssim 1,$$

where $c_1 = 1, c_2 = J^2$ and $c_d = 2^{(d-2)J}$ for $d \ge 3$.

For the SSC iterative method, we have

Theorem 8.8.55 *The Algorithm 5.2.13 with the subspaces $\mathcal{V}_k$ given by* (8.8.44) *satisfies*

$$\|E_J\|_A^2 \le 1 - \frac{2-\omega_1}{Cc_d}$$

provided that the R_k satisfy (8.8.53) *with $\omega_1 < 2$.*

Compared with the usual multigrid method, the smoothing in the SSC hierarchical basis method is carried out only on the set of new nodes $\mathcal{N}_k \setminus \mathcal{N}_{k-1}$ on each subspace $\mathcal{M}_k$. The method proposed by Bank, Dupont and Yserentant [?] can be viewed as such an algorithm with R_k given by an appropriate Gauss–Seidel iteration. Numerical examples in [?] show that the hierarchical basis SSC algorithm converges much faster than the corresponding SSC algorithm.

8.8-e Relation with BPX preconditioners

Observing that S_k in (8.8.48) is a submatrix of $\mathcal{I}_k$ given in (8.7.38), we then have

$$(\mathcal{H}\alpha, \alpha) \le (\mathcal{B}\alpha, \alpha), \quad \forall\, \alpha \in \mathbb{R}^n.$$

In view of the above inequality, if we take

$$\hat{\mathcal{H}} = \sum_{k=0}^{J-1} h_k^{2-d} \mathcal{S}_k \mathcal{S}_k^t + I,$$

we obtain

$$(\mathcal{H}\alpha, \alpha) \le (\hat{\mathcal{H}}\alpha, \alpha) \le (\mathcal{B}\alpha, \alpha), \quad \forall\, \alpha \in \mathbb{R}^n.$$

Even though $\hat{H}$ appears to be a very slight variation of H, numerical experiments have shown a great improvement over H for $d = 2$. We refer to Xu and Qin [?] for the numerical results.

8.9 Locally refined grids

In practical computations, finite element grids are often locally refined (by using some error estimators or other adaptive strategies). In this subsection, we shall describe optimal multigrid procedures for adaptive grids. Our presentation here is based on [?], [?] and [?].

With appropriate rearrangement and grouping, we may assume that the mesh refinement can be done in the following fashion. We first start with the original domain Ω which is also denoted by Ω_0. We introduce a relatively coarse and quasi-uniform triangulation of Ω_0 with a mesh size h_0 and denote the corresponding finite element space by $\mathcal{V}_0 \subset H_0^1(\Omega_0)$. Let Ω_1 be a subregion where we wish to increase the resolution: we do so by subdividing the elements of the first triangulation to get a new triangulation of Ω_1 with mesh size h_1 in Ω_1 and we introduce an additional finite element space $\mathcal{V}_1 \subset H_0^1(\Omega_1)$. We repeat this process and finally get a collection of subdomains Ω_i together with the corresponding finite element spaces $\mathcal{V}_i$ defined on a triangulation of mesh size h_i for $i = 1, 2, \cdots, J$ for some integer $J > 1$. Throughout, we have

$$\Omega_i \subset \Omega_{i-1}, \quad \mathcal{V}_{i-1} \cap H_0^1(\Omega_i) \subset \mathcal{V}_i \subset H_0^1(\Omega_i), \quad i = 1, 2, \cdots, J.$$

The finite element space on the repeatedly refined mesh can be written as

$$\mathcal{V} = \sum_{i=0}^{J} \mathcal{V}_i.$$

The only restrictions on the mesh domains $\{\Omega_k\}$ are that $\partial\Omega_k$ for $k \geq 1$ consists of edges of mesh triangles in the mesh $\mathcal{T}_{k-1}$ and that there is at least one edge from $\mathcal{T}_{k-1}$ contained in Ω_k.

Operators $A_i : \mathcal{V}_i \mapsto \mathcal{V}_i$ and $Q_i, P_i : \mathcal{V} \mapsto \mathcal{V}_i$ can be defined as before. If we choose $R_i : \mathcal{V}_i \mapsto \mathcal{V}_i$ to be some appropriate approximate solvers of the A_i, we then have all the ingredients to define our PSC and SSC algorithms. In this setting, the coarse space $\mathcal{M}_0$ may not be very coarse: we therefore assume that, for the PSC type algorithm, the first subspace solver R_0 is SPD and satisfies

$$(R_0^{-1}v, v) \eqsim (A_0 v, v) \quad \forall\, v \in \mathcal{M}_0, \tag{8.9.56}$$

and for the SSC type of algorithm, we assume that R_0 satisfies

$$\|(I - R_0 A_0)\|_A \leq \delta_0 \tag{8.9.57}$$

for some $\delta_1 \in (0, 1)$ independent of mesh parameters.

As for the other subspace solvers R_k for $k > 1$, we assume for clarity that R_k is given by a Gauss–Seidel iteration or properly damped Jacobi iteration. Apparently other reasonable solvers can also be adopted.

We would like to remark that the corresponding PSC and SSC methods in this setting can be viewed as a 'nested' multigrid method associated with the multilevel spaces given by

$$\mathcal{M}_k = \sum_{i=0}^{k} \mathcal{V}_i, \quad 0 \le k \le J,$$

but with special coarse space solvers R_k only defined on the subspace $\mathcal{V}_k$, namely the smoothings are only carried out in the refined regions.

8.9-a PSC version

Let us first consider the preconditioner corresponding to the PSC algorithm:

$$B = \sum_{k=0}^{J} R_k Q_k. \tag{8.9.58}$$

Thanks to Lemma **??**, as for the quasi-uniform case, we can use our general framework in §5 to obtain the following theorem.

Theorem 8.9.59 *If R_0 satisfies* (8.9.56) *and R_k are given by Jacobi or symmetric Gauss–Seidel, then the PSC preconditioner* (8.9.58) *yields a uniformly bounded condition number*

$$\kappa(BA) \eqsim 1.$$

As a special example, we may choose R_k as follows:

$$R_k v = h_k^{2-d} \sum_{x_i^k \in \mathcal{N}_k} (v, \phi_i^k)\phi_i^k. \tag{8.9.60}$$

As we know, this corresponds to Richardson iteration which is equivalent to Jacobi iteration. For such a choice, we obtain that

$$Bv = R_0 v + \sum_{k=1}^{J} h_k^{2-d} \sum_{x_i^k \in \mathcal{N}_k} (v, \phi_i^k)\phi_i^k. \tag{8.9.61}$$

We notice that (8.9.61) is exactly the preconditioner given in [**?**] for locally refined meshes.

The hierarchical basis type algorithms for these composite grids can be obtained by the decomposition with $\mathcal{V}_i = (I_i - I_{i-1})\mathcal{V}$ (here $I_i : \mathcal{V} \mapsto \mathcal{M}_i$ is the nodal value interpolation operator). It is easy to see that the SSC type preconditioner is

$$Hv = R_0 v + \sum_{k=1}^{J} h_k^{2-d} \sum_{x_i^k \in \mathcal{N}_k \setminus \mathcal{N}_{k-1}} (v, \phi_i^k)\phi_i^k. \tag{8.9.62}$$

This preconditioner is equivalent to what is given in [**?**] for refined meshes. We further point out the corresponding algorithm in [**?**] for the refined meshes is the SSC algorithm by choosing R_k to be some appropriate Gauss–Seidel iteration.

SSC version

The SSC version corresponds to multigrid algorithms with smoothings done in the refined regions. Similarly we have the following convergence theorem.

Theorem 8.9.63 *If R_0 satisfies* (8.9.57) *and R_k are given by properly damped Jacobi or Gauss–Seidel, then the corresponding SSC iteration yields a uniform contraction:*

$$\|I - BA\|_A \leq \delta$$

for some $\delta \in [\delta_0, 1)$ independent of mesh parameters.

Additional bibliographic comments The multilevel algorithm for finite element or finite difference equations was first developed in the early sixties by the Russian mathematician Fedorenko [**?**]. In the early seventies, Brandt [**?**] brought this method to the attention of western countries and extensive research has been done on this method since then. Nowadays it has become one of the most popular and powerful iterative methods.

Multilevel methods for composite grids can be traced back to Brandt [**?**] or to composite grids in McCormick [**?**] (see also the references therein). The finite element space on a mesh refined in this way is $\mathcal{V} = \sum_{i=0}^{J} \mathcal{V}_i$. PSC and SSC methods can be naturally obtained with Gauss–Seidel iterations as subspace solvers. The SSC iteration corresponds to a multigrid method with smoothings carried out only on the refined region discussed in Brandt [**?**]. The PSC preconditioner was first considered in Bramble, Pasciak and Xu [**?**].

The aforementioned refined grids may not give the minimal degrees of freedom from an approximation point of view, but they are a computationally efficient approach and have the desirable structure for multigrid applications. If more traditional graded meshes are used on each level, proper nested subspaces are then hard to come by and the corresponding nonnested multigrid methods are more complicated (cf. Zhang [**?**]).

Chapter 9

Two-grid methods for Non SPD problems

The main purpose of this chapter is to present some discretization techniques based on two (or more) finite element subspaces for solving partial differential equations (PDE). Examples under our study here for this technique are linear non SPD second order elliptic boundary value problems.

It is also observed that that the the finite element solution of a nonsymmetric and/or indefinite linear equation is "super-close" to the finite element solution of some symmetric positive definite equation. These facts are useful for both the theoretical analysis and the design of efficient solvers for the resultant algebraic systems. In fact, we shall present several algorithms based on such considerations for nonsymmetric and/or indefinite linear systems. For technical reasons for some of the results , we will further assume the domain is polygonal, convex or of Lipschitz continuous boundary.

9.1 Finite element approximation for general SPD problems

Let the elliptic operator L have the form

$$Lu = -\sum_{i,j=1}^{N} \frac{\partial}{\partial x_j}(a_{ij}\frac{\partial u}{\partial x_i}) + cu,$$

satisfying: $a_{ij}, \in W^{1,\infty}(\Omega), 0 \leq c \in L^{\infty}(\Omega)$, and $A = (a_{ij})$ is uniformly positive definite on Ω.

Define

$$a(u,v) = \int_{\Omega} \sum_{i,j=1}^{N} a_{ij} \frac{\partial u}{\partial x_i} \frac{\partial v}{\partial x_j},$$

then there exists a constant $C > 0$ such that

$$a(w,w) \geq C^{-1} \int_\Omega |\nabla w|^2, \quad \forall w \in H^1(\Omega). \tag{9.1.1}$$

The variational formulation is: Find $u \in H_0^1(\Omega)$ such that

$$a(u,v) = (f,v), \quad \forall v \in H_0^1(\Omega). \tag{9.1.2}$$

By Lax-Milgram Lemma, one sees that this variational problem has a unique solution.

Let

$$V_h = \{v \in H_0^1(\Omega) : v \text{ is continuous and piecewise linear w.r.t.} T_h\}$$

associated with the partition T_h. Define the finite element approximation scheme by: Find $u_h \in V_h$ such that

$$a(u_h,v) = (f,v), \quad \forall v \in V_h. \tag{9.1.3}$$

Theorem 9.1.4 *There holds*

$$\|u - u_h\|_0 + h|u - u_h|_1 \leq Ch^2|u|_2. \tag{9.1.5}$$

Proof. Frist of all, we have the following identity

$$a(u - u_h, v) = 0, \quad \forall v \in V_h, \tag{9.1.6}$$

from which we get, for any $v \in V_h$

$$a(u_h - v, \phi) = a(u - v, \phi), \quad \forall \phi \in V_h. \tag{9.1.7}$$

Hence, (9.1.1) leads to

$$|u - u_h|_1 \leq C(|u - v|_1 + |u_h - v|_1) \leq C|u - v|_1, \quad \forall v \in V_h,$$

or

$$|u - u_h|_1 \leq C \inf_{v \in V_h} |u - v|_1 \leq Ch|u|_2.$$

For the L^2-norm error estimate, we use the the so-called Aubin-Nitsche trick.

Recall the regularity estimate, there exists $w \in H_0^1(\Omega) \cap H^2(\Omega)$ satisfying

$$a(w, \phi) = (u - u_h, \phi), \quad \forall \phi \in H_0^1(\Omega) \tag{9.1.8}$$

and

$$\|w\|_2 \leq C\|u - u_h\|_0. \tag{9.1.9}$$

Thus, we have

$$\begin{aligned}\|u-u_h\|_0^2 &= a(w,u-u_h) = \inf_{v\in V_h} a(w-v,u-u_h)\\ &\le C\inf_{v\in V_h}|w-v|_1|u-u_h|_1\\ &\le Ch^2|w|_2|u|_2\\ &\le Ch^2|u|_2\|u-u_h\|_0,\end{aligned}$$

and

(9.1.10) $$\|u-u_h\|_0 \le Ch^2|u|_2.$$

This completes the proof.

Remark. The MG method for this problem is similar to that of Poisson equations.

9.2 Convection-diffusion problems

We consider the following non SPD equation

$$\hat{L}u = -\sum_{i,j=1}^{N}\frac{\partial}{\partial x_j}(a_{ij}\frac{\partial u}{\partial x_i}) + \sum_{i=1}^{N} b_i\frac{\partial u}{\partial x_i} + cu,$$

where $b_i \in W^{1,\infty}(\Omega)$.

Set

$$N(u,v) = \int_\Omega \sum_{i=1}^{N} b_i\frac{\partial u}{\partial x_i}v + cuv$$

and

$$\hat{a}(u,v) = a(u,v) + N(u,v).$$

We shall use the following variational formulation: Find $u \in H_0^1(\Omega)$ such that

(9.2.11) $$\hat{a}(u,v) = (f,v), \quad \forall v \in H_0^1(\Omega).$$

Theorem 9.2.12 *This problem is uniquely solvable if*

(9.2.13) $$\inf_{u\in H_0^1(\Omega)}\sup_{v\in H_0^1(\Omega)}\frac{\hat{a}(u,v)}{|u|_1|v|_1} > 0 \quad \textit{and} \quad \inf_{v\in H_0^1(\Omega)}\sup_{u\in H_0^1(\Omega)}\frac{\hat{a}(u,v)}{|u|_1|v|_1} > 0,$$

especially if $c \ge 0$.

Remark. One sees that The $inf - sup$ conditions are satisfied if

$$\hat{a}(w,w) \geq C^{-1}|w|_1^2, \quad \forall w \in H_0^1(\Omega).$$

In the finite element method, the discrete problem is always SPD (hence solvable) for SPD problem. However, for a non-SPD problem, the discrete problem:

Find $u_h \in V_h$ such that

$$\hat{a}(u_h, v) = (f, v), \quad \forall v \in V_h. \tag{9.2.14}$$

is not always solvable although the original PDE is uniquely solvable.

Theorem 9.2.15 *Assume that the continuous problem is uniquely solvable, namely the $inf - sup$ conditions are satisfied, then the discrete finite element problem is also uniquely solvable provided h is sufficently small. Furthermore*

$$\inf_{v \in V_h} \sup_{\varphi \in V_h} \frac{\hat{a}(v,\varphi)}{|v|_1 |\varphi|_1} > 0 \quad \textit{and} \quad \inf_{v \in V_h} \sup_{\varphi \in V_h} \frac{\hat{a}(\varphi, v)}{|v|_1 |\varphi|_1} > 0. \tag{9.2.16}$$

Proof. Let us frist note that the discrete $inf - sup$ conditions implies the finite element equation is uniquely solbable.

Let $P_h : H_0^1(\Omega) \longrightarrow V_h$ satisfy

$$a(w - P_h w, v) = 0, \quad \forall v \in V_h.$$

Then

$$\begin{aligned} \hat{a}(v, P_h w) &= \hat{a}(v,w) - \hat{a}(v, w - P_h w) \\ &= \hat{a}(v,w) - (a - \hat{a})(v, w - P_h w) \\ &\geq \hat{a}(v,w) - C^{-1}|v|_1 \|w - P_h w\|_0 \\ &\geq \hat{a}(v,w) - C^{-1} h |v|_1 |w|_1. \end{aligned}$$

The proof of the first estimate then follows by using the fact that

$$|P_h w|_1 \leq C|w|_1.$$

The proof of the second estimate is similar.

Theorem 9.2.17 *There holds*

$$\|u - u_h\|_0 + h|u - u_h|_1 \leq C h^2 |u|_2. \tag{9.2.18}$$

Proof. From the identity

$$\hat{a}(u - u_h, v) = 0, \quad \forall v \in V_h, \tag{9.2.19}$$

we get, for any $v \in V_h$

$$\hat{a}(u_h - v, \phi) = \hat{a}(u - v, \phi), \quad \forall \phi \in V_h. \tag{9.2.20}$$

Hence, the discrete $inf - sup$ conditions lead to

$$|u - u_h|_1 \le C(|u - v|_1 + |u_h - v|_1) \le C|u - v|_1, \quad \forall v \in V_h,$$

or

$$|u - u_h|_1 \le C \inf_{v \in V_h} |u - v|_1 \le Ch|u|_2.$$

For the L^2-norm error estimate, we also use the the Aubin-Nitsche technique.

Recall the regularity estimate, there exists $w \in H_0^1(\Omega) \cap H^2(\Omega)$ satisfying

$$\hat{a}(\phi, w) = (u - u_h, \phi), \qquad \forall \phi \in H_0^1(\Omega) \tag{9.2.21}$$

and

$$\|w\|_2 \le C\|u - u_h\|_0. \tag{9.2.22}$$

Thus, we have

$$\begin{aligned} \|u - u_h\|_0^2 &= \hat{a}(u - u_h, w) = \inf_{v \in V_h} \hat{a}(u - u_h, w - v) \\ &\le C \inf_{v \in V_h} |w - v|_1 |u - u_h|_1 \\ &\le Ch^2 |w|_2 |u|_2 \\ &\le Ch^2 |u|_2 \|u - u_h\|_0, \end{aligned}$$

and

$$\|u - u_h\|_0 \le Ch^2 |u|_2. \tag{9.2.23}$$

This completes the proof.

9.3 Two-grid methods

For the convection operator $N : Nv \equiv \hat{L}v - Lv$, here are some observations:

1. $''\hat{L} >> N''$ on high frequencies, and $L^{-1}N$ is small on high frequencies.

2. Lower frequencies can be resolved efficicently by coarse grid.

We consider two finite element spaces $V_H \subset V_h$, V_H, V_h is associated with the coarse grid of size H and the finer grid of size h, respectively.

Let us now present our first two-grid algorithm.

Algorithm I

1. Find $u_H \in V_H$ such that

$$\hat{a}(u_H, v) = (f, v), \quad v \in V_H$$

.

2. Correct the residual on finer space V_h: Find $u^h \in V_h$ such that

$$a(u^h, v) + N(u_H, v) = (f, v), \quad v \in V_h.$$

We note that the linear system in the second step of the above algorithm is SPD.

Theorem 9.3.24 *Assume $u^h \in V_h$ is the solution obtained by Algorithm I for H is sufficiently small, then*

$$\|u_h - u^h\|_1 \le CH^2 |u|_2$$

and

$$\|u - u^h\|_1 \le C(h + H^2)|u|_2$$

provided that $u \in H_0^1(\Omega) \cap H^2(\Omega)$.

Proof. Let $u_H = \hat{P}_H u$. A direct calculation shows that

$$\begin{aligned} a(u_h - u^h, v) &= -N((I - \hat{P}_H)\, u_h, v) \\ &\le C\|(I - \hat{P}_H)u_h\| \, \|v\|_1 \\ &\le C(H\|u - u_h\|_1 + \|(I - \hat{P}_H)u\|) \, \|v\|_1 \\ &\le CH^2 \, \|u\|_2 \, \|v\|_1. \end{aligned}$$

The desired result then follows.

Remark. Although $\|u_h - u^h\|_1 = O(h + H^2)$, $\|u_h - u^h\|_0 = O(h + H^2)$ is optimal.

Algorithm II

1. Find $u_H \in V_H$ such that

$$\hat{a}(u_H, v) = (f, v), \quad v \in V_H$$

.

2. Correct the residual on finer space V_h: Find $u^h \in V_h$ such that

$$a(u^h, v) + N(u_H, v) = (f, v), \quad v \in V_h$$

.

3. Coarse grid correction: Find $e^H \in V_H$ such that

$$\hat{a}(e^H, \varphi) = (f, \varphi) - \hat{a}(u^h, v), \quad \varphi \in V_H$$

Theorem 9.3.25 *For Algorithm II, there holds*

(9.3.26) $$\|u_h - (u^h + e^H)\|_0 \le CH(h + H^2)|u|_2.$$

Proof. From the identity

$$(I - \hat{P}_H(u_h - u^h) = u_h - (u^h + e^H)$$

and

$$\|(I - \hat{P}_H)w\|_0 \le CH\|w\|_1,$$

we immediately obtain the result.

Remark. One possible MG method for correction diffusion problem is that only the leading SPD term is to do smoothing on the finer grids (except the coarsest grid where on exact solver can be used)

.

Algorithm III Let $u_h^0 = 0$.
1. Find $e_H \in V_H$ such that

$$\hat{a}(e_H + u_h^k, v) = (f, v), \quad v \in V_H$$

2. Find $u_h^{k+1} \in V_h$ such that

$$a(u_h^{k+1}, v) + N(u_h^k + e_H, v) = (f, v), \quad v \in V_h$$

.

Theorem 9.3.27 *Assume $u_h^k \in V_h$ is the solution obtained by Algorithm III for $k \ge 1$, then*

$$\|u_h - u_h^k\|_1 \le CH^{k+1}\|u\|_2,$$

and

$$\|u - u_h^k\|_1 \le C(h + H^{k+1})\|u\|_2$$

Proof By definition ,

$$\begin{aligned} a(u_h - u_h^k, v) &= N((I - \hat{P}_H)(u_h^{k-1} - u_h), v) \\ &\le C\|(I - \hat{P}_H)(u_h^{k-1} - u_h)\| \, \|v\|_1 \\ &\le CH \, \|u_h^{k-1} - u_h\|_1 \, \|v\|_1. \end{aligned}$$

This implies

$$\|u_h - u_h^k\|_1 \le CH\|u_h - u_h^{k-1}\|_1, \quad k = 1, 2, \cdots$$

Applying the above estimate successively yields

$$\|u_h - u_h^k\|_1 \le CH^{k-1}\|u_h - u_h^1\|_1 \le CH^{k+1}\|u\|_2.$$

This completes the proof.

Chapter 10

The EAFE scheme and CWS method for convection-domainated problems

In this chapter, we discuss a special finite element scheme for convection diffusion problems. The special discretization technique, known as *edge-average finite element*(EAFE is short) method, is obtained by properly averaging the PDE coefficients on element edges. When it is applied to convection diffusion equations in conservative form (in any spatial dimensions), the resulting finite element stiffness matrix is an M-matrix under some mild assumption for the underlying finite element grids. Thus the method is particularly effective for convection dominated problems.

10.1 Introduction

We shall consider the following convection-domainated problems

$$(10.1.1)\qquad \begin{cases} \dfrac{\partial u}{\partial t} + \operatorname{div}\,(\beta u) - \varepsilon \triangle u &= f, \quad x \in \Omega \\ u &= 0, \quad x \in \partial\Omega, \end{cases}$$

where $\varepsilon << 1$ and $\beta = (\beta_1, \cdots, \beta_N)$ is some smooth vector.

Associated with the stationary problem (with the limiting case $\varepsilon = 0$)

$$(10.1.2)\qquad \begin{cases} \beta \cdot \nabla u + \gamma u &= f, \quad x \in \Omega, \\ u &= g, \quad x \in \Gamma_-, \end{cases}$$

where $\Gamma_- = \{x \in \Omega : \beta \cdot \mathbf{n} < 0\}$ is the in flow boubdary, the characteristics are $\{X_i : 1 \leq i \leq N\}$ defined by

$$\left\{ \begin{array}{rcl} \dfrac{dX_i}{ds} & = & \beta_i(s), \quad 1 \leq i \leq N, \\ X_i(0) & = & \bar{X} \end{array} \right. \tag{10.1.3}$$

or

$$\left\{ \begin{array}{rcl} \dfrac{d\mathbf{X}}{ds} & = & \beta(s), \\ \mathbf{X}(0) & = & \bar{\mathbf{X}}. \end{array} \right. \tag{10.1.4}$$

On a characteristics

$$\frac{d}{ds}(u(\mathbf{X}(s))) = \nabla u \cdot \frac{d\mathbf{X}}{ds} = \beta \cdot \nabla u,$$

we have

$$\left\{ \begin{array}{rcl} \dfrac{du(s)}{ds} + \gamma u(\mathbf{X}(s)) & = & f(\mathbf{X}(s)), \\ u(\mathbf{X}(0)) & = & u(\bar{\mathbf{X}}). \end{array} \right. \tag{10.1.5}$$

Thus, if $u = g$ is given on Γ_-, then the 1^{st} order hyperbolic problem (1.2) is uniquely solvable.

10.2 One dimensional example

For the model problem

$$\left\{ \begin{array}{l} -\varepsilon u'' + u' = 0, \quad 0 < x < 1, \\ u(0) = 1, \quad u(1) = 0, \end{array} \right. \tag{10.2.6}$$

the exact solution is

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

Now we use the central finite difference scheme

$$u''(x_i) \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + O(h^2),$$

$$u'(x_i) \approx \frac{u_{i+1} - u_{i-1}}{2h} + O(h^2),$$

and obtain a finite difference scheme

$$\left\{ \begin{array}{l} -\varepsilon \dfrac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + \dfrac{u_{i+1} - u_{i-1}}{2h} = 0, \\ u_0 = 1, \quad u_N = 0. \end{array} \right. \tag{10.2.7}$$

The finite difference solution is:

$$u_i = \frac{\lambda^i - \lambda^N}{1 - \lambda^N},$$

where $\lambda = \frac{2\varepsilon + h}{2\varepsilon - h}$.

One sees that if $\varepsilon < h/2$, then $\lambda < 0$, which implies u_i oscillates.

Now we turn to use another so-called upwinding scheme by using $u'(x_i) \approx \dfrac{u_i - u_{i-1}}{h}$:

$$\left\{ \begin{array}{l} -\varepsilon \dfrac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + \dfrac{u_i - u_{i-1}}{h} = f_i, \\ u_0 = 1, \quad u_N = 0. \end{array} \right. \tag{10.2.8}$$

Let $\nu = \dfrac{2\varepsilon}{h}$, then $\nu > 0$ and

$$-\nu u_{i+1} + (1 + 2\nu) u_i - (1 + \nu) u_{i-1} = h f_i, \tag{10.2.9}$$

which is equivalent to the algebraic system

$$A\mu = \beta, \tag{10.2.10}$$

where

$$A = \text{diag}(-(1+\nu), 1 + 2\nu, -\nu), \quad \beta = h f_i.$$

One has that A is an M-matrix, i.e., any entry of A^{-1} is nonnegative. Note that $A \approx D - L$, the lower triangular, Gauss-Seidel method is very efficient!

Conclusion:

- The central difference scheme for the convection term treats to oscillates numerical solution if h is not sufficiently small;
- The upwinding finite difference scheme is much better and it gives rise to an M-matrix as coefficient matrix;
- The Gauss-Seidel iteration always converges for M-matrix system but the convergence depends crucially on the ordering of the unknowns.

10.3 Multidimensional convection-dominated problems

Consider the simple situation

$$\left\{ \begin{array}{rcl} -\varepsilon \Delta u + u_x & = & f, \ x \in \Omega, \\ u & = & 0, \ x \in \partial\Omega. \end{array} \right.$$

A corresponding upwinding scheme is as follows

$$-\varepsilon\frac{u_{i+1,j}+u_{i-1,j}+u_{i,j+1}+u_{i,j-1}-4u_{i,j}}{h^2}+\frac{u_{i,j}-u_{i-1,j}}{h}=f_{i,j}.$$

More complicated problems are

$$\begin{cases} -\nabla\cdot(\alpha\nabla u+\beta u) &= f,\ x\in\Omega,\\ u &= 0,\ x\in\partial\Omega, \end{cases}$$

where $\mid \beta/\alpha \mid >> 1$ and α,β is variable.

To introduce the so-called edge-averaged finite element (EAFE in short) scheme, we consider the Poisson equation

$$\begin{cases} -\Delta u &= f,\ x\in\Omega,\\ u &= 0,\ x\in\partial\Omega. \end{cases}$$

Given a triangulation $\mathcal{T}_h$, let $V_h\subset H_0^1(\Omega)$ be the piecewise linear finite element space and

$$A=((\nabla\phi_i,\nabla\phi_j))$$

is the stiffness matrix.

Under what condition is A an M-matrix? It is proved that in two dimensional case, a necessary and sufficient condition is that $\mathcal{T}_h$ is a Delauney tringulation, namely, the sum of two opositive angles of any edge is not greater than π.

Given $T\in\mathcal{T}_h$, we introduce the following notation:

- q_j $(1\le j\le n+1)$: the vertices of T;
- E_{ij} or simply E: the edge connecting two vertices q_i and q_j;
- F_j: the $(n-1)$ dimensional simplex opposite to the vertex q_j;
- θ_{ij}^T or θ_E^T: the angle between the faces F_i and F_j;
- κ_E^T: $F_i\cap F_j$, namely the $n-2$ dimensional simplex opposite to the edge E;
- $\delta_E\phi=\phi(q_i)-\phi(q_j)$, for any continuous function on $E=E_{ij}$;
- $\tau_E=\delta_E x=q_i-q_j$, a directional vector of E.

We shall denote the nodes in $\mathcal{T}_h$ by x_j $j=1,\ldots,N_h$. This is a "global" notation for all the vertices on the grid. Thus, we shall use $q_j\,(=q_j^T)$, $j=1,\ldots,n+1$ in a fixed element $T\in\mathcal{T}_h$ and x_j $(j=1,\ldots N_h)$ for all the nodes. The edges E_{ij} will denote either the edge (q_i,q_j) in an element T, or the edge (x_i,x_j) living somewhere on the grid. This slight abuse of notation should not be a source of confusion.

Let (a_{ij}^T) be the element stiffness matrix on T. Then, for u_h, $v_h\in V_h$, we have

$$\int_T\nabla u_h\cdot\nabla v_h dx=\sum_{i,j}a_{ij}^T u_h(q_i)v_h(q_j). \tag{10.3.11}$$

By $a_{ii}^T = -\sum_{j\neq i} a_{ij}^T$, we can easily obtain the following simple, but important identity

(10.3.12)
$$\int_T \nabla u_h \cdot \nabla v_h dx = -\sum_{i<j} a_{ij}^T (u_h(q_i) - u_h(q_j))(v_h(q_i) - v_h(q_j)),\ u_h, v_h \in V_h.$$

We can rewrite the bilinear form in the following way:

$$\int_\Omega \nabla u_h \cdot \nabla v_h dx = \sum_{T\in\mathcal{T}_h} \sum_{E\subset T} \omega_E^T \delta_E u_h \delta_E v_h. \tag{10.3.13}$$

where $\omega_E^T = -a_{ij}^T$ with E connecting the vertices q_i and q_j.

Note that for a constant vector $J_T = \nabla u_h$, there holds

$$\delta_E u_h = |E| \frac{\partial u_h}{\partial \tau_E} = |E| \nabla u_h \cdot \frac{\tau_E}{|E|} = J_T \cdot \tau_E,$$

which implies that for any constant vector J_T,

$$\int_T J_T \cdot u_h = \sum_{E\subset T} \omega_E^T \delta_E u_h \delta_E v_h \delta_E v_h. \tag{10.3.14}$$

To present the main idea more clearly, we shall first derive the discrete scheme for a simplified model problem with simplified assumptions. We shall discuss more general case later. Specifically we consider:

$$\begin{cases} \mathcal{L}u \equiv -\nabla \cdot (\alpha(x)\nabla u + \beta(x)u) = f(x) & x \in \Omega, \\ u = 0 & x \in \partial\Omega. \end{cases} \tag{10.3.15}$$

We assume that $\alpha \in C^1(\bar\Omega)$ with $0 < \alpha_{min} \le \alpha(x) \le \alpha_{max}$ for every $x \in \Omega$, $\beta \in \left(C^0(\bar\Omega)\right)^2$, and $f \in L^2(\Omega)$.

The weak formulation of the problem (10.3.15) is: find $u \in H_0^1(\Omega)$ such that

$$a(u,v) = f(v), \qquad \text{for every } v \in H_0^1(\Omega), \tag{10.3.16}$$

where

$$a(u,v) = \int_\Omega (\alpha(x)\nabla u + \beta(x)u) \cdot \nabla v dx, \quad f(v) = \int_\Omega f(x) v dx. \tag{10.3.17}$$

It can be shown that (10.3.16) is uniquely solvable and there exists a constant $c_0 > 0$ such that for every $v \in H_0^1(\Omega)$

$$\sup_{\phi\in H_0^1(\Omega)} \frac{a(\phi,v)}{\|\phi\|_{1,\Omega}} \ge c_0 \|v\|_{1,\Omega}; \qquad \sup_{\phi\in H_0^1(\Omega)} \frac{a(v,\phi)}{\|\phi\|_{1,\Omega}} \ge c_0 \|v\|_{1,\Omega}. \tag{10.3.18}$$

Set $J \equiv J(u) = \alpha(x)\nabla u + \beta(x)u$, then

$$\int_\Omega J \cdot \nabla v = \int_\Omega fv, \quad \forall v \in H_0^1(\Omega).$$

One sees that

$$J\tau_E = \alpha\nabla u \cdot \tau_E + (\beta\tau_E)u = \alpha|\tau_E|\frac{\partial u}{\partial \tau_E} + \beta_E u,$$

$$\frac{\partial u}{\partial \tau_E} + \frac{\beta\tau_E}{\alpha|\tau_E|}u = \frac{1}{\alpha|\tau_E|}J \cdot \tau_E.$$

Thus, if ψ_E satisfies

$$\frac{\partial \psi_E}{\partial \tau_E} = \frac{\beta\tau_E}{\alpha|\tau_E|},$$

then we have

$$e^{-\psi_E}\frac{\partial(e^{\psi_E}u)}{\partial \tau_E} = \frac{1}{|\tau_E|}\alpha^{-1}(J(u) \cdot \tau_E) \tag{10.3.19}$$

and

$$\frac{\partial}{\partial \tau_E}(e^{-\psi_E}u) = \frac{1}{|\tau_E|}\int_E \alpha^{-1}e^{-\psi_E}(J(u) \cdot \tau_E) \tag{10.3.20}$$

Now, if $J(u) \approx J_T(u)$ is constant vector, then

$$J_T(u)\tau_E \approx (\frac{1}{|\tau_E|}\int_E \alpha^{-1}e^{-\psi_E})^{-1}\delta_E(e^{\psi_E}u)$$

and hence

$$\int_\Omega J(u)\nabla v_h = \sum_{T\in\mathcal{T}_h}\int_T J(u)\nabla v_h \approx \sum_T \int_T J_T(u)\nabla v_h$$

$$= \sum_{T\in\mathcal{T}_h}\sum_{E\subset T}\omega_E^T J_T\tau_E\delta_E v_h$$

$$\approx \sum_{T\in\mathcal{T}_h}\sum_{E\subset T}\omega_E^T(\frac{1}{|\tau_E|}\int_E \alpha^{-1}e^{-\psi_E})^{-1}\delta_E(e^{\psi_E}u)\delta_E v_h$$

$$= \sum_{E\in\mathcal{T}_h}(\sum_{T\supset E}\omega_E^T)(\frac{1}{|\tau_E|}\int_E \alpha^{-1}e^{-\psi_E})^{-1}\delta_E(e^{\psi_E}u)\delta_E v_h$$

$$= \sum_{E\in\mathcal{T}_h}\omega_E(\frac{1}{|\tau_E|}\int_E \alpha^{-1}e^{-\psi_E})^{-1}\delta_E(e^{\psi_E}u)\delta_E v_h,$$

where $\omega_E = \sum_{T\subset E}\omega_E^T = -\sum_{T\supset E}a_{ij}^T = -a_{ij} = -(\nabla\phi_i, \nabla\phi_j)$ and $\sum_{T\supset E}$ means summation that takes all simplexes T containing E.

Set

$$a_h(u_h, v_h) = \sum_{E\in\mathcal{T}_h} \omega_E\Big(\frac{1}{|\tau_E|}\int_E \alpha^{-1}e^{-\psi_E}\Big)^{-1}\delta_E(e^{\psi_E}u)\delta_E v_h,$$

we define a finite element scheme, called EAFE scheme, by: Find $u_h \in V_h$ such that

$$a_h(u_h, v_h) = (f, v_h), \quad \forall v_h \in V_h. \tag{10.3.21}$$

The stiffness matrix from the EAFE scheme is an M-matrix if and only if the matrix $((\nabla\phi_i, \nabla\phi_j))$ for the Poisson equation is an M-matrix. In this case, EAFE scheme is an upwinding scheme. Gauss-Seidel iteration always converges and the rate of convergence depends crucially on the ordering of unknowns.

For a flow without recirculation, there exists optimal ordering algorithm that leads to rapidly convergent Gauss-Seidel iteration (Tarjon algorithm in graph theory).

Chapter 11

Stokes equations

The stationary Stokes equation for an incompressible Newtwonian fluid with viscosity μ, enclosed in the domain $\Omega \in \mathbb{R}^2$, and acted upon by a volume load f can be characterised by the following minimization problem:

$$\inf_{v\in V, \text{div } v=0} \{\mu \int_\Omega |e(v)|^2 dx - \int_\Omega f \cdot v dx\}.$$

Here

$$V = (H^1_D(\Omega))^n, \quad (n = 2, 3).$$

Let p be the Lagrange multiplier associated with the incompressibility constraint, the above minimization problem can be written as the following variational problems:

$$\left\{ \begin{array}{rcll} a(u,v) + b(v,p) & = & (f,v) & \forall\, v \in V, \\ b(u,q) & = & 0 & \forall\, q \in W, \end{array} \right. \tag{11.0.1}$$

where

$$a(u,v) = 2\mu \int_\Omega e(u) : e(v)\, dx, \quad b(v,q) = -\int_\Omega q \text{div } v\, dx.$$

In the strong form, the above variatinal problem can be written as

$$\left\{ \begin{array}{rcll} -\mu\Delta u + \nabla p & = & f & \text{in } \Omega; \\ -\text{div } u & = & 0 & \text{in } \Omega; \\ u & = & 0 & \text{on } \Gamma_D \\ \mu e(u)n + pn & = & 0 & \text{on } \Gamma_T. \end{array} \right. \tag{11.0.2}$$

Let

$$B = -div : V \mapsto W, \quad B^t = \nabla : W \mapsto (H^{-1}(\Omega))^n.$$

Then the Stokes equation can be formally written as follows:

$$\begin{pmatrix} A & B^t \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}. \tag{11.0.3}$$

For simplicity, in the rest of the paper, we assume that $\Gamma_D = \partial\Omega$ and define

$$V = (H_0^1(\Omega))^n, W = \{q \in L^2(\Omega) : \int_\Omega q = 0\}$$

with norms given by

$$|v|_{1,\Omega} = \|\nabla v\|_{0,\Omega}, \quad |q|_{0,\Omega} = \|q\|_{o,\Omega}$$

respectively.

11.1 Well-posedness and inf-sup condition

Formally, let us consider an LU factorizatin of the operator matrix for the Stokes eqation:

$$\begin{pmatrix} A & B^t \\ B & 0 \end{pmatrix} = \begin{pmatrix} I & 0 \\ -BA^{-1} & I \end{pmatrix} \begin{pmatrix} A & B^t \\ 0 & -S \end{pmatrix}$$

where the Schur compliment operator S is given by

$$S = BA^{-1}B^t : W \mapsto W. \tag{11.1.4}$$

Therefore the Stokes operator is an isomorphism if and only the Schur compliment operator S is an isomorphism. Apparently S is a bounded operator. It is also not difficult to see that S is onto. Now the question if S is invertible and has a bounded inverse, namely if there is a constant $c > 0$ such that

$$|Sq|_0 \ge c|q|_0 \quad \forall\, q \in W.$$

But

$$|Sq|_0 = \sup_{\phi\in W} \frac{(S\phi, q)}{|\phi|_0} \eqsim \sup_{v\in V} \frac{b(v,q)}{|v|_1}$$

since for $v \in V$ such that $-\Delta v = \nabla\phi$, we have $|v|_1 \eqsim |\phi|_0$.

Therefore S is an isomorphism if there exists conatant $\alpha > 0$ such that

$$\inf_{q\in W} \sup_{v\in V} \frac{b(v,q)}{|v|_1 |q|_0} \ge \alpha > 0. \tag{11.1.5}$$

This is known as an inf-sup condition. This condition can be indeed proved rigorously. The details of the proof is omitted here.

11.2 Finite element discretization

In the discretization of Stokes equations, we need a pair of finite element subspaces $V_h \subset V$ and $W_h \subset W$.

$$(11.2.6) \qquad \left\{ \begin{array}{rcll} a(u_h, v_h) + b(v_h, p_h) & = & (f, v_h) & \forall\, v_h \in V_h, \\ b(u_h, q_h) & = & 0 & \forall\, q_h \in W_h. \end{array} \right.$$

Let

$$B = -\text{div}_{\,h} : V_h \mapsto W_h, \quad B^t = \nabla : W_h \mapsto (H^{-1}(\Omega))^n.$$

Then the Stokes equation can be written in a matrix form

$$(11.2.7) \qquad \begin{pmatrix} A_h & B_h^t \\ B_h & 0 \end{pmatrix} \begin{pmatrix} u_h \\ p_h \end{pmatrix} = \begin{pmatrix} f_h \\ 0 \end{pmatrix}.$$

Like in the continuous case, the discrete Stokes equation is well-posed if the inf-sup condition is satisfied

$$(11.2.8) \qquad \inf_{q_h \in W_h} \sup_{v_h \in V_h} \frac{b(v_h, q_h)}{|v_h|_1 |q_h|_0} \geq \alpha > 0.$$

If we can choose α independently of mesh size h, we then call the pair $V_h \times W_h$ to be a stable element. If this case, the discrete Stokes equation is well-posed, in certain sense, uniformly with respect to the mesh size h.

One immediate consequence of the above inf-sup condition is the following best approximation property.

Lemma 11.2.9 *Let $u \in V$ be divergence-free:* div $u = 0$. *Then, under the inf-sup condition* (11.2.8)

$$\inf_{w_h \in V_h(0)} |u - w_h|_1 \leq (1 + \frac{1}{\alpha}) \inf_{v_h \in V_h} |u - v_h|_1.$$

Here $V_h(0)$ is the discrete divergence-free space:

$$V_h(0) = \{v_h \in V_h : \ b(v_h, q_h) = 0 \quad \forall\, q_h \in W_h.$$

Proof. Given $v_h \in V_h$, let $r_h \in V_h$ be such that

$$b(r_h, q_h) = b(u - v_h, q_h) \quad \forall\, q_h \in W_h.$$

□

Theorem 11.2.10 *For a pair of stable element $V_h \times W_h$, namely the inf-sup condition* (11.1.5) *is satisfied uniformly with respect to h, then*

$$(11.2.11) \qquad |u - u_h|_1 + |p - p_h|_0 \leq C(\sup_{v_h \in V_h} |u - v_h|_1 + \inf_{q_h \in W_h} |p - q_h|_0)$$

for some constant C independent of h.

Proof. The proof is simple □

(11.2.12) EXERCISE. Prove the best approximation property (11.2.11) holds uniformly with respect to h (if and) only if the inf-sup condition (11.2.8) holds uniformly with respect to h.

The verification of the discrete inf-sup condition is not an easy task in general. Let us describe one convenient technique in this direction.

Lemma 11.2.13 *The discrete inf-sup condition is uniformly satisfied if there is a linear operator* $\Pi_h : V \mapsto V_h$ *such that*

$$b(v - \Pi_h v, q_h) = 0 \quad \forall\, q_h \in W_h \tag{11.2.14}$$

and

$$|\Pi_h v|_1 \le c|v|_1 \quad \forall\, v \in V \tag{11.2.15}$$

for some constant c *independent of* h*.*

The proof of this lemma is straightforward by using the inf-sup condition (11.1.5) on the continuous level.

(11.2.16) EXERCISE. Prove Lemma 11.2.13.

$P_2 - P_0$ elements and Crouzeix-Raviart elements One seemingly natural pair of elements is continuous piecewise P_1 (for velocity) versus piecewise constant (for pressure). But it is easy to see that this pair of element is not good. In fact, we have seen before, for some special grid, the only divergence free function is the zero function, hence the inf-sup condition can not be satisfied.

To fix this problem, we can enrich the velocity space. We shall now discuss two such enrichments. With the piecewise constant functions still used for the pressure space, the first one is to use P_2 element for velocity and the second one is to use nonconforming P_1 element for velocity (Crouzeix-Raviart element).

We shall now prove that the $P_2 - P_0$ is indeed a stable element. We proceed to verify the conditions (11.2.14) and (11.2.15). For any $v \in V$, we need to define $v_h = \Pi_h v$ satisfying (11.2.14) which, in the present situation, reads, for each element T

$$\int_T \mathrm{div}\,(v - v_h) = \int_{\partial T} (v - v_h) \cdot n = 0.$$

This last condition can obviously be satisfied if we construct v_h in such a way that, for each edge e_i of T,

$$\int_{e_i} v_h = \int_{e_i} v, \quad i = 1, 2, 3. \tag{11.2.17}$$

The above condition can be easily satisfied by defining the value of v_h at the midpoint of each e_i using the relation (11.2.17). But since we are in P_2 we also need to define v_h on the nodal points as well. The procedure we will take is first to define the nodal values of v_h and then define the midpoint values of v_h by (11.2.17). To do this we use make use the average interpolant introduced in §**??** which we denote here by Π_1. We then define

$$\Pi_h v = \Pi_1 v + \Pi_2(v - \Pi_1 v) \tag{11.2.18}$$

where $\Pi_2 : V \mapsto V_h$ is defined locally such that, for each $v \in V$, $\Pi_2 v$ is zero on each vertex and its value of the midpoint of any edge e is determined by a relation like (11.2.17)

$$\int_e \Pi_2 v = \int_e v.$$

It is not difficult to see that Π_h constructed in this way satisfies all the desired properties.

(11.2.19) EXERCISE. Prove that the Π_h defined by (11.2.18) satisfies both (11.2.14) and (11.2.15).

As we see from the above analysis, as far as inf-sup condition is concerned, the digree of freedoms on all the nodal points in the P_2 element is not necessary. So let us just throw them away! The resulting element is then a nonconforming piecewise linear element which is continuous at all midpoints. This element is known as the Crouzeix-Raviart element. Indeed, for purely Dirichlet boundary condition, inf-sup condition can be verified easily. But, unfortunately, this element seems to only work for the pure Dirichet boundary condition for sure since the discrete Korn's inequality may fail to hold for other boundary conditions. Since the Stokes equation with a purely Dirichlet boundary condition is of very little practical importance, the significance of this pair of element is then very limited.

11.3 Some examples of stable elements

In this section, we describe some examples of stable elements.

11.3-a No bubbles

Taylor-Hood elements These elements refer to the following

- P_2^0–P_1^0 for both $\mathbb{R}^2$ (triangle) and $\mathbb{R}^3$ (tedrahedral).
- P_3^0–P_2^0 for both $\mathbb{R}^2$ (triangle) and $\mathbb{R}^3$ (tedrahedral).
- Q_2^0–Q_1^0 for $\mathbb{R}^2$ (square).

Scott-Vogelius elements It is known that the $P_k^0 - -P_{k-1}^{-1}$ are not stable for $k \leq 3$. But Scott-Vogelius proved that

- P_k^0–P_{k-1}^{-1} for $\mathbb{R}^2$ (triangle) and $k \geq 4$

is indeed stable for triangular grid with mild restrictions.

11.3-b Stablization by bubble functions

A bubble function is a function that is defined locally on each element and has support on that element. The most commonly used bubble function is the cubic bubble

$$b_3(x) = \lambda_1(x)\lambda_2(x)\lambda_3(x).$$

MINI-element The MINI-element is the known stable element with the minimal degree of freedom and it consists of

- $P_1^0 + \{b_3\} - P_1^0$

Crouzeix-Raviart elements To stablize the $P_2^0 - P_1^{-1}$ element, Crouzeix and Raviart added a bubble to the velocity space:

- $P_2^0 + \{b_3\} - P_1^{-1}$.

11.4 An abstract theory for saddle problem and mixed method

Given two Banach spaces V and W and $f \in V^*$ and $g \in W^*$, we shall study the following variational problem

$$\text{(11.4.20)} \quad \text{Find } (u,p) \in V \times W, \quad \left\{ \begin{array}{rcll} a(u,v) + b(v,p) & = & < f, v > & v \in V \\ b(u,q) & = & < g, q > & q \in W. \end{array} \right.$$

Associated with the bilinear form a and b, we define two linear operator $A : V \mapsto V^*$ and $B : V \mapsto M^*$ by

$$\text{(11.4.21)} \qquad \langle Au, v\rangle = a(u,v) \quad \forall\, u, v \in V$$

$$\text{(11.4.22)} \qquad \langle Bv, q\rangle = b(v,q) \quad \forall\, v \in V, q \in W.$$

Let B^* be the dual operator of B, namely

$$\text{(11.4.23)} \qquad \langle B^*q, v\rangle = \langle q, Bv\rangle = b(v,q) \quad \forall\, v \in V, q \in W.$$

Then the variational problem can written as

$$\text{(11.4.24)} \qquad \text{Find } (u,p) \in V \times W, \quad \left\{ \begin{array}{rcll} Au + B^*p & = & f & \text{in } V^* \\ Bu & = & g & \text{in } W^*. \end{array} \right.$$

Let us now study the well-posedness of the above problem. One obvious necessary condition is that

$$B \text{ is surjective}$$

$\Longleftrightarrow$

$$B^* \text{ is injective and has a closed range}$$

$\Longleftrightarrow$

$$\inf_{q\in W}\frac{\|B^*q\|}{\|q\|_W}\equiv\beta>0$$

$\Longleftrightarrow$

$$\inf_{q\in W}\sup_{v\in V}\frac{b(v,q)}{\|v\|_V\|q\|_W}\equiv\beta>0. \tag{11.4.25}$$

In summary we have

$$B \text{ is surjective} \Longleftrightarrow \inf_{q\in W}\sup_{v\in V}\frac{b(v,q)}{\|v\|_V\|q\|_W}\equiv\beta>0. \tag{11.4.26}$$

A simple consequence of the equivalence (11.4.26) is the following result due to Babuska.

Theorem 11.4.27 *The variational problem, with $g\in W^*$,*

$$\textit{Find } u\in V,\quad b(u,q)=\langle g,q\rangle \quad \forall\, q\in W$$

is well-posed if and only if

$$\inf_{q\in W}\sup_{v\in V}\frac{b(v,q)}{\|v\|_V\|q\|_W}>0,\quad \inf_{v\in V}\sup_{q\in W}\frac{b(v,q)}{\|v\|_V\|q\|_W}>0. \tag{11.4.28}$$

Proof. The said variational problem is well-posed means that B is an isomorphism. By (11.4.26), the first inf-sup condition in (11.4.28) means that B is sujective while the second inf-sup condition in (11.4.28) means that B^* is sujective. The desired result then follows. □

It is not hard to make the equivalence (11.4.26) in a little bit more precise fashion. Let us introduce some additional notation:

$$Z=\{v\in V:\ b(v,q)=0\quad \forall\, q\in W\},$$

and

$$Z^0=\{f\in V^*:\ \langle f,v\rangle=0\quad \forall\, v\in V\}.$$

Lemma 11.4.29 *The following properties are equivalent:*
1. the inf-sup condition holds:

$$\inf_{q\in W}\sup_{v\in V}\frac{b(v,q)}{\|v\|_V\|q\|_W}\equiv\beta>0; \tag{11.4.30}$$

2. the operator B^* is an isomorphism from W onto Z^0 and

$$\|B^*q\|_{V^*}\geq\beta\|q\|_W\quad \forall\, q\in W; \tag{11.4.31}$$

3. the operator B is an isomorphism from $Z^\perp$ onto W^* and

$$\|Bv\|_{W^*}\geq\beta\|v\|_V\quad \forall\, v\in Z^\perp. \tag{11.4.32}$$

Lemma 11.4.33 *Assume that* (11.4.30) *holds. Then, for any* $f \in V^*$ *and* $g \in W^*$, $(u,p) \in V \times W$ *solves* (11.4.20) *if any only* $u = u_g + u_0$, *with* $u_g \in Z^{\perp}, u_0 \in Z$, *and* $p \in W$ *solve the following decoupled problems*

$$u_g \in Z^{\perp}, \quad b(u_g, q) = < g, q > \quad \forall\, q \in W, \tag{11.4.34}$$

$$u_0 \in Z, \quad a(u_0, v) = < f, v > - a(u_g, v) \quad \forall\, v \in Z \tag{11.4.35}$$

$$p \in W, \quad b(v, p) = < f, v > - a(u, v) \quad \forall\, v \in V. \tag{11.4.36}$$

Thanks to Lemma 11.4.29, both (11.4.34) and (11.4.36) are well-posed. The well-posedness of (11.4.35) falls into the framework of Theorem 11.4.27. Thus we have the main theorem of this section.

Theorem 11.4.37 *The variational problem* (11.4.20) *is well-posed if and only if the following three inf-sup conditions are satisfied*

$$\inf_{u \in Z} \sup_{v \in Z} \frac{a(u,v)}{\|u\|_V \|v\|_V} \equiv \alpha > 0, \quad \inf_{v \in Z} \sup_{u \in Z} \frac{a(u,v)}{\|u\|_V \|v\|_V} = \alpha > 0, \tag{11.4.38}$$

and

$$\inf_{q \in W} \sup_{v \in V} \frac{b(v,q)}{\|v\|_V \|q\|_W} \equiv \beta > 0. \tag{11.4.39}$$

11.5 Uzawa method and agumented Lagrange method

Eliminating the variable u from the first equation, we have the equation for p as follows

$$Sp = \tilde{g}, \quad \text{with } S = BA^{-1}B^*, \tilde{g} = A{-}^1 f - g.$$

By the inf-sup condition S is well-conditioned, hence it is reasonable to use a simple Richardson iteration:

$$p^{n+1} = p^n + \omega(\tilde{g} - Sp^n). \tag{11.5.40}$$

Here ω is a positive parameter and as we know the above iteration converges if we choose

$$0 < \omega < 2/\rho(S). \tag{11.5.41}$$

Set $u^{n+1} = A^{-1} - A^{-1}B^*p^n$, then we have the following so-called Uzawa algorithm:

$$Au^{n+1} + B^*p^n = f \tag{11.5.42}$$

$$p^{n+1} = p^n + \omega(g - Bu^{n+1}). \tag{11.5.43}$$

These two equations are just an alternative way of writing the simple Richardson iteration (11.5.40). The convergence behavior of this type of iteration is well-known. One problem about this iteration is that we need to choose the damping parameter to satisfy (11.5.41), which is not always easy. One remedy of this problem is to use some penalty method.

Let us first write the original mixed method by the following equivalent way:

$$\begin{aligned}(A + rB^*B)u + B^*p &= f + rB^*g \\ Bu &= g.\end{aligned} \tag{11.5.44}$$

Here r is any positive number. One interesting property of this alternative form is that its Schur compliment matrix

$$S_r = B(A + rB^*B)^{-1}B^*$$

has a known upper bound for its spectral radius, for any $r > 0$:

$$\rho(S_r) < 1/r.$$

In fact, a simple calculation (by using the definition of the eigenvalue) shows that

$$\sigma(S_r) = \{\frac{\mu}{1 + r\mu} : \mu \in \sigma(S)\}$$

and, hence

$$\rho(S_r) = \frac{1}{r + (\rho(S))^{-1}}$$

As a result, the corresponding Richardson iteration:

$$p^{n+1} = p^n + \omega(\tilde{g}_r - S_r p^n). \tag{11.5.45}$$

converges as long as

$$0 < \omega \le 2r.$$

By a well-known property of the Richardson iteration, an optimal choice of ω is as follows

$$\omega_{\rm opt} = \frac{2}{\lambda_{\min}(S_r) + \lambda_{\max}(S_r)} = \frac{2(1 + r\mu_0)(1 + r\mu_1)}{\mu_0 + \mu_1 + 2r\mu_0\mu_1},$$

where

$$\mu_0 = \lambda_{\min}(S), \quad \mu_1 = \lambda_{\max}(S).$$

For large r, we have $\omega_{opt} \approx r$. Thus a practical choice of ω is

$$\omega = r.$$

With such a choice of ω, we have

$$\rho(I - rS_r) = \frac{1}{1 + \mu_0 r}.$$

Hence

$$\|p - p^n\| \le \frac{1}{(1 + \mu_0 r)^n}\|p - p^0\|.$$

In summary, we have the following iterative algorithm:

Algorithm 11.5.46

$$(A + rB^*B)u^{n+1} + B^*p^n = f + rB^*g \tag{11.5.47}$$

$$p^{n+1} = p^n + r(g - Bu^{n+1}). \tag{11.5.48}$$

Theorem 11.5.49 *The iterates from the augmented Lagrange iteration admit the following estimate*

$$\|p-p^n\| \le \frac{1}{(1+\mu_0 r)^n}\|p-p^0\|, \quad \|u-u^n\|_A \le \frac{1}{\sqrt{r}}\|p-p^n\| \le \frac{1}{\sqrt{r}}\frac{1}{(1+\mu_0 r)^n}\|p-p^0\|.$$

Proof. The estimate for $p-p^n$ has been estimated above. To estimate $u-u^n$, we use the following identity

$$u - u^n = -(A + rB^*B)^{-1}B^*(p - p^n).$$

It follows that

$$\begin{aligned} (A(u-u^n), u-u^n) &= (A(A+rB^*B)^{-1}B*(p-p^n), (A+rB^*B)^{-1}B*(p-p^n)) \\ &\le (B*(p-p^n), (A+rB^*B)^{-1}B*(p-p^n)) \\ &= (p-p^n, B(A+rB^*B)^{-1}B*(p-p^n)) \\ &\le \frac{1}{r}(p-p^n, (p-p^n)). \end{aligned}$$

The desired estimate then follows. □

We note that the convergence of this iteration is faster for larger r. It would then appears to be desirable to choose very large r. As we know, there is no free lunch, a large r must cause some other part of the algorithm more difficult. In fact, in this case, the matrix $A + rB^*B$ becomes more difficult to solve for larger r.