Multi-Grid Algorithms for Elliptic Boundary-Value Problems

Craig Carl Douglas[1]

Technical Report #223

May, 1982

[1]Department of Computer Science, Yale University, New Haven, CT 06520

This research was supported in part by ONR Grant N00014-76-C-0277, FSU-ONR Grant F1N00014-80-C-0076, NSF MCS-8104874, Yale University.

ABSTRACT

Multi-Grid Algorithms for Elliptic Boundary-Value Problems

Craig Carl Douglas

Yale University, 1982

This dissertation is primarily concerned with solving the large sparse linear systems which arise in connection with finite-element or finite-difference procedures for solving self-adjoint elliptic boundary-value problems. These procedures can be expressed in terms of abstract variational problems on Hilbert spaces, which we can solve iteratively. Our (multi-grid) schemes involve a sequence of auxiliary finite-dimensional spaces which do not have to be nested. We want to approximate the solution using the largest (finite-dimensional) space. These schemes are recursive in nature: they combine smoothing iterations in a space with solving correction problems in smaller spaces. Under certain circumstances, the solution to a problem can be approximated well using smaller spaces. Since the smaller spaces are required to have geometrically fewer unknowns than the largest space, the savings in computation can be substantial. In fact, we prove that these procedures are optimal order under appropriate conditions. This theory is discretization independent and can be applied to problems which do not arise from partial differential equations.

As examples, we consider three particular discretizations of variable coefficient self-adjoint second order elliptic boundary-value problems. The first is a finite-element discretization on a convex domain in two dimensions. The second is a finite-difference discretization in one dimension. The last is a finite-difference discretization on the unit square. For the finite-difference discretizations, we present a summary of sharper results that can be proved when the number of grids is restricted to two. The proofs can be found in Douglas [19]. We first investigate how the order of interpolation affects the rate of convergence. We restrict our attention to the cases of piecewise-linear and piecewise-cubic interpolation between grids. This can be used to optimize the number of smoothing iterations and correction recursions to minimize the amount of work required.

We describe the implementation issues involved with a finite-difference multi-grid FORTRAN code for general second order (possibly nonself-adjoint) elliptic problems in rectangular domains. The code uses more general multi-grid schemes than those defined initially. Complexity bounds and numerical experiments are included.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Motivation

In this dissertation, we discuss the iterative solution of large sparse linear systems which arise in connection with finite-element or finite-difference procedures for solving self-adjoint elliptic boundary-value problems. Suppose we use finite-differences to discretize a problem to get a sparse system of linear equations of the form

$$Au = F, \tag{1.1}$$

where A is symmetric positive definite. For simplicity, assume that we are solving a problem whose domain is the unit square, which we denote by Ω with boundary $\partial\Omega$. A uniform mesh Ω^h with mesh spacing h can be laid on Ω. Then (1.1) can be rewritten as

$$\begin{aligned} A^h u^h &= h^2 F^h \\ &= f^h. \end{aligned} \tag{1.2}$$

One way of solving for u^h in (1.2) is by relaxation. For instance, consider Gauss-Seidel [50]:

1. Generate an initial guess u^h_0.
2. Compute u^h_{i+1} from u^h_i by

   $$u^h_{i+1} = [\ f^h - L^h u^h_{i+1} - K^h u^h_i\ ] / D^h,$$

   where L^h, K^h, and D^h are the lower, upper, and diagonal parts of A^h.
3. Repeat step 2 until convergence.

A drawback to Gauss-Seidel is that its convergence rate is unacceptably slow. Even for simple problems like Poisson's equation with Dirichlet boundary conditions on the unit square, the convergence rate is only $1 - O(h^2)$.

The number of iterations needed for acceptable convergence might be decreased by choosing a better initial guess. Suppose we have two grids Ω^{qh} and Ω^h, where $q > 1$ is an integer. The linear system

$$A^{qh}u^{qh} = f^{qh}$$

can be solved faster than

$$A^h u^h = f^h.$$

Once u^{qh} has been solved for, we can use it to generate an initial guess for u^h. In fact, Southwell [46] proposed the following algorithm in 1940:

<u>Algorithm</u> <u>1.1</u> (Southwell): (a) Solve the problem on Ω^{qh} by any means available.

(b) Interpolate u^{qh} onto Ω^h as the initial guess to u^h.

(c) Perform relaxation sweeps on Ω^h until convergence.

Although it was probably a known method at the time, this appears to be the first published multi-grid algorithm.

Another possible method of reducing the number of iterations of Gauss-Seidel is to compute corrections occasionally on a coarser grid. In 1961, the Russian mathematician Federenko [22] proposed the first iterative multi-grid algorithm. It also uses two grids, but computation begins on Ω^h. Grid Ω^{qh} is used only to solve for corrections to u^h in a manner similar to iterative improvement. The algorithm is as follows:

<u>Algorithm</u> <u>1.2</u> (Federenko): (a) Generate an initial guess for u^h.
(b) Do n relaxation sweeps on Ω^h.
(c) Compute residuals on Ω^h and "inject" them into Ω^{qh} as the the right-hand side. By this we mean compute

$$r^h = f^h - A^h u^h,$$

then transfer the residuals at the points coincident between the two grids:

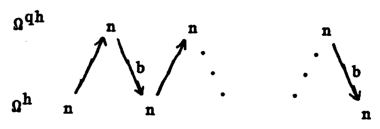
$$f^{qh} = q^2 r^h.$$

(The factor of q^2 above is there because $(qh)^2$ is factored into f^{qh} instead of h^2.)
(d) Starting with an initial guess of zero for e^{qh}, solve the residual correction problem

$$A^{qh} e^{qh} = f^{qh}$$

to moderate accuracy using relaxation sweeps.
(e) Interpolate the solution e^{qh} onto Ω^h and add it to u^h.
(f) Repeat steps (b) - (e) until convergence.

We can represent Algorithm 1.2 schematically by



b = bilinear interpolation
n = do n relaxation sweeps

Note that if we use zero as the initial guess to u^h and do no relaxation sweeps before solving only one residual correction problem, we get Southwell's Algorithm.

Federenko [22, 23] analyzed the convergence rate of Algorithm 1.2 for the model problem of the Poisson equation on the unit square. Eigenfunctions of A^h were classified as "good" and "bad". Good eigenfunctions included ones that were smooth on Ω^h and had few sign changes:, bad ones changed signs often and oscillated rapidly. It is a well known fact that relaxation methods tend to annihilate the bad error components rapidly, but not the good ones. However, good eigenfunctions on Ω^h are bad eigenfunctions on Ω^{qh}. By using a coarser mesh, we attempt to solve a new problem consisting of bad eigenfunctions which can be annihilated quickly.

When $q = 2$, Federenko [23] proved that his algorithm required O(NlogN) operations for his model problem, where N is the rank of A^h. Federenko also pointed out that the coarse grid problem could be considered the fine grid in another two-level scheme, leading to the first recursive multi-grid algorithm definition. In 1966, the Russian mathematician Bakhvalov [8] extended Federenko's results to a finite-difference discretization of a general linear second order elliptic boundary-value problem. He also showed that the operation count is O(N) asymptotically in the number of levels used.

In 1971, Astrakhantsev [4] proved the same convergence result as Bakhvalov, but for a finite-element discretization. C^0-piecewise linear polynomials were used as basis functions on triangular elements. The domain assumed was two-dimensional, bounded, and simply connected. The operation count derived for the multi-grid scheme was not optimal, however.

In the finite-difference area, Nicolaides [39] analyzed the Dirichlet Poisson equation in a square in 1975. Brandt [15] published a comprehensive, but not rigorous, work in 1977. Brandt uses local mode (Fourier) analysis to investigate smoothing rates and grid transfers. Recently, work in the area of problems with discontinuous coefficients has been done by Alcouffe, Brandt, Dendy, Hyman, and Painter [3, 17] and Kettler and Meijerink [35]. The use of multi-grid combined with intermediate interpolation grids has been investigated by Foerster, Stuben, and Trottenberg [24].

Finite-element analysis continued in 1977 when

Hackbusch [28, 29, 30] proved optimal order results for a general linear second order elliptic boundary-value problem. At about the same time, Bank and Dupont [9] and Nicolaides [37, 38] independently proved results similar to those of Hackbusch, but by different techniques. Van Rosendale [49] extended the results of Bank and Dupont to the case of locally refined grids.

Recently, multi-grid has been investigated as a preconditioning method for the conjugate gradient method. Axelsson and Gustafsson [5] and Kettler and Meijerink [35] have used multi-grid based on (modified) incomplete decomposition preconditioning methods.

1.2 Outline of the Dissertation

In Chapter 2, we discuss the iterative solution of an abstract elliptic variational problem. Our schemes involve a sequence of auxiliary finite-dimensional spaces which do not have to be nested. We want to approximate the solution using the largest (finite-dimensional) space. These schemes are recursive in nature: they combine smoothing iterations in a space with solving correction problems in smaller spaces. Under certain circumstances, the solution to a problem can be approximated well using smaller spaces. Since the smaller spaces are required to have geometrically fewer unknowns than the largest space, the savings in computation can be substantial. In fact, we prove that these procedures are optimal order under appropriate conditions. While this theory is applied to solving the large sparse linear systems which arise in connection with finite-element or finite-difference procedures for solving self-adjoint elliptic boundary-value problems, it can be applied to problems which do not arise from partial differential equations.

In Chapters 3 - 5, we prove that the multi-grid schemes analyzed in Chapter 2 converge for three particular discretizations of variable coefficient self-adjoint second order elliptic boundary-value problems. The first is a finite-element discretization on a convex domain in two dimensions. The second is a finite-difference discretization in one dimension. The last is a finite-difference discretization on the unit square.

At the end of Chapters 4 and 5, we present a summary of sharper results that can be proved when the multi-grid algorithms of Chapter 2 are restricted to two grids. The proofs can be found in Douglas [19]. We first investigate how the order of interpolation affects the rate of convergence. We restrict our attention to the cases of piecewise-linear and piecewise-cubic interpolation between grids. This can be used to optimize the number of smoothing iterations and correction recursions to minimize the amount of work required.

In Chapter 6, we describe the implementation issues involved with a finite-difference multi-grid FORTRAN code for general second order (possibly nonself-adjoint) elliptic problems in rectangular domains. The code uses more general multi-grid schemes than those defined in Chapter 2. Implementation details and complexity bounds are included. Numerical experiments and conclusions are included in Chapter 7.

CHAPTER 2
General Theory

In this chapter, we discuss the iterative solution of an abstract elliptic variational problem. Our scheme involves a sequence of finite-dimensional spaces M_j, $j = 1, 2, \ldots, k$. We want to approximate the solution using the largest space. Under certain circumstances, the solution to a problem can be approximated well using smaller spaces. Since we require the smaller spaces to have geometrically fewer unknowns than the largest one, the savings in computation can be substantial. In fact, we prove that these procedures are optimal order under appropriate conditions. While this theory is applied to solving the large sparse linear systems which arise in connection with finite-element or finite-difference procedures for solving self-adjoint elliptic boundary-value problems in Chapters 3 - 5, it can also be applied to problems which do not arise from partial differential equations.

Our k-level scheme is related to the multi-grid techniques used by Bank and Dupont [9], which is related to the techniques of Brandt [15], Bakhvalov [8], Federenko [22, 23], Nicolaides [38, 39], and Hackbusch [28, 29, 30]. The earlier proofs are for particular discretizations of model elliptic boundary-value problems. Their domains are covered by meshes or triangulations which are refined uniformly. Only Van Rosendale's proof [49] allows nonuniformly refined domains. The proofs here use abstract function space arguments which make no reference to the particular discretization, domain, or method of refinement. Further, we do not require the solution spaces to be nested as in the proofs of the cited references.

Assume we are given a triple,

$$\{H,\ a(u,v),\ f(v)\}, \tag{2.1}$$

where H is a Hilbert space with norm $\|\cdot\|$, $a(u,v)$ is a continuous symmetric bilinear form on $H \times H$, and $f(v): H \to R$ is a continuous linear functional. Furthermore, we assume that there exists a constant $a_0 > 0$ such that

$$a(v,v) \geq a_0\|v\|^2 \text{ for all } v \in H.$$

The bilinear form $a(\cdot,\cdot)$ induces the <u>energy norm</u>

$$|||u|||^2 = a(u,u).$$

We seek an approximation to the solution of

<u>Problem 2.1</u>: Given $\{H, a(u,v), f(v)\}$, find $u \in H$ such that

$$a(u,v) = f(v) \text{ for all } v \in H.$$

Problem 2.1 has a unique solution (see Ciarlet [16]).

We now consider the finite-dimensional approximation of

Problem 2.1. Let M_j, $j \geq 1$, be a sequence of N_j-dimensional spaces. Associated with each space M_j is a continuous, symmetric, positive-definite bilinear form $a_j(u,v)$ and a continuous, bounded linear form $f_j(v)$. For some $\sigma \geq 2$, we require that

$$N_j \sim \sigma N_{j-1}. \tag{2.2}$$

We will see that σ is important: when $\sigma > 2$, we can construct optimal order algorithms to approximate the solution to Problem 2.1.

We assume that linear operators exist which project H onto M_j and inject M_j into H for any $j \geq 1$:

$$p_j: H \xrightarrow{\text{onto}} M_j \quad \text{and}$$
$$i_j: M_j \xrightarrow{\text{1-1}} H. \tag{2.3}$$

For $j \geq 1$, the linear operators defined by

$$E_j: M_{j-1} \to M_j \quad \text{and}$$
$$R_j: M_j \xrightarrow{\text{onto}} M_{j-1} \tag{2.4}$$

interpolate between adjacent solution spaces. One definition of E_j and R_j is

$$E_j = p_j i_{j-1} \quad \text{and} \quad R_j = p_{j-1} i_j. \tag{2.5}$$

A natural extension is to define R_j as the adjoint of E_j:

$$R_j = E_j^*.$$

Figure 2-1: Space Operators

$$\begin{tikzcd}
 & \vdots \\
 & M_{j-1} \arrow[dl, bend right, "i_{j-1}"'] \arrow[d, shift right, "E_j"'] \\
H \arrow[ur, bend left=10, "p_{j-1}"] \arrow[r, shift right, "p_j"'] & M_j \arrow[l, shift right, "i_j"'] \arrow[u, shift right, "R_j"']
\end{tikzcd}$$

In this case,

$$E_j^* = i_{j-1}^* p_j^* = p_{j-1} i_j = R_j.$$

If we assume that $p_j = i_j^*$, then all of the operators can be defined in terms of the injection operators i_j. However, we will have occasion to use more general operators than those in (2.5). Figure 2-1 shows the relationship between these operators and the various spaces.

The finite-dimensional approximation of Problem 2.1 is

<u>Problem 2.2</u>: Given $\{M_j, a_j(u,v), f_j(v)\}$, find $u_j \in M_j$ such that

$$a_j(u_j,v) = f_j(v) \quad \text{for all } v \in M_j.$$

Associated with each space M_j are eigenvalues $\lambda_i^{(j)}$ and eigenfunctions (eigenvectors) $\xi_i^{(j)}$, $1 \le i \le N_j$, satisfying

$$a_j(v,\xi_i^{(j)}) = \lambda_i^{(j)} (v,\xi_i^{(j)})_j \quad \text{for all } v \in M_j,$$

where $(\cdot,\cdot)_j$ denotes the inner product in M_j. Let δ_{ik} be the Kronecker delta. Without loss of generality,

$$\begin{aligned}
& 0 < \lambda_1^{(j)} \le \lambda_2^{(j)} \le \dots \le \lambda_{N_j}^{(j)} = \Lambda_j, \\
& (\xi_i^{(j)},\xi_k^{(j)})_j = \delta_{ik}, \quad 1 \le i,k \le N_j, \quad \text{and} \qquad (2.6) \\
& a_j(\xi_i^{(j)},\xi_k^{(j)}) = \lambda_i^{(j)}\delta_{ik}.
\end{aligned}$$

With each space M_j, we define discrete norms

$$|||v|||_s^2 = \sum_{i=1}^{N} c_i^2 (\lambda_i^{(j)})^s, \qquad \text{for } v = \sum_{i=1}^{N} c_i \xi_i^{(j)},$$

where we have suppressed the j subscript on the norm and $-2 \le s \le 2$. Note that $|||v|||_1 = |||v|||$ is the usual energy norm on level j. Hereafter, we drop both the superscripts from the eigenvalues and eigenfunctions (eigenvectors) and the subscript from the dimension of the spaces.

We require bilinear forms of adjacent spaces to have a particular relationship. As a consequence, the energy norms on two adjacent spaces are uniformly consistent.

<u>Hypothesis 2.3</u> (Energy Norm Consistency): Let $j > 1$ be an integer. Then there exists a positive constant C_1, independent of j, such that

$$a_j(E_j v, E_j w) = C_1 \, a_{j-1}(v,w), \text{ for all } v,w \in M_{j-1}.$$

Recall that $|||v|||^2 = a_j(v,v)$. For any $v \in M_{j-1}$, this hypothesis implies that the energy norm of v on level j-1 is equal to a constant times the energy norm of E_jv on level j.

We now define and analyze a k-level iterative procedure for solving Problem 2.2. The process involves solving problems like Problem 2.2 sequentially for j=1, 2,... , k. The <u>k-level</u> <u>scheme</u> has three parameters: m and n, which determine the number of smoothing iterations used; and p, which is used in a recursion iteration.

<u>Algorithm</u> <u>2.4</u>: Given an integer $k > 0$ and $\{M_j,\ a_j(\cdot,\cdot),\ f_j(\cdot)\}_{j=1}^{k}$, we want to approximate $u_k \in M_k$, where $a_k(u_k,v) = f_k(v)$ for all $v \in M_k$.

(a) If k = 1, then solve directly.

(b) If k > 1, then one iteration of the k-level scheme takes an initial guess $z_0 \in M_k$ to a final approximation $z_{m+n+1} \in M_k$ in three steps:

(i) if $n > 0$, define z_i, $1 \le i \le n$, by

$$(z_i - z_{i-1}, v)_k \;=\; \Lambda_k^{-1}[f_k(v) - a_k(z_{i-1},v)], \text{ for all } v \in M_k. \qquad (2.7)$$

(ii) Let $q \in M_{k-1}$ be the approximation of $\bar{q} \in M_{k-1}$ obtained by applying p iterations of the (k-1)-level scheme to the residual equation

$$a_{k-1}(\bar{q},v) \;=\; C_1^{-1}\,\{f_k(E_kv) - a_k(z_n,E_kv)\} \qquad (2.8)$$

$$\;=\; \bar{f}_{k-1}(v), \text{ for all } v \in M_{k-1},$$

starting from an initial guess zero. Then set

$$z_{n+1} \;=\; z_n + E_kq. \qquad (2.9)$$

(iii) If $m > 0$, then define z_i, $n+2 \le i \le m+n+1$, by (2.7).

Figure 2-2: Three-Level Example of Algorithm 2.4

```
Two iterations on level three, p = 2

Level

  1       ds   ds        ds   ds
         /  \ /  \      /  \ /  \
  2     n   n+m   m    n   n+m   m
       /           \  /           \
  3   n             n+m            m

 ds = direct solve
n,m = number of smoothing iterations
```

In the <u>correction</u> <u>recursion</u> <u>iteration</u> (step (ii)), we approximately compute the elliptic projection of the error in M_{k-1} using p iterations of the (k-1)-level scheme applied to a problem of Problem 2.2's form with j = k - 1. In the <u>smoothing</u> <u>iterations</u> (steps (i) and (iii)), error components whose oscillation are "large" are damped. A simultaneous displacement procedure is used in this step. Later in this chapter we will see that Λ_k^{-1} can be replaced by a particular type of bound (see Hypothesis 2.6). We will also see that (2.7) can be replaced by other iterations which are computationally more attractive, but do not affect the character of our convergence results. The use of this iteration simplifies the initial analysis of the convergence. Figure 2-2 contains a three-level, two-iteration example of Algorithm 2.4 with p = 2. Note that computation begins with the largest space and uses the smaller ones only to solve correction recursion problems.

There are three cases of note in Algorithm 2.4: (a) when n > 0 and $m = 0$, (b) when $n = 0$ and $m > 0$, and (c) when $n > 0$ and $m > 0$. Case (a) is the scheme analyzed by Astrakhantsev [4], Bank and Dupont [9], Hackbusch [28, 29, 30], Nicolaides [37, 38], and Van Rosendale [49] for finite-element discretizations of various elliptic boundary-value problems. Federenko [22, 23] and Bakhvalov [8] analyzed this case for finite-difference discretizations. Brandt [15] analyzed (nonrigorously) cases (b) and (c) for finite-difference discretizations using local mode analysis. The motivation for studying cases (b) and (c) comes from trying to understand the behavior of a large finite-difference program (which is described in Chapter 6). It was observed empirically that case (b) sometimes required fewer correction recursions to achieve the same accuracy as case (a).

Before proving a convergence theorem for Algorithm 2.4, we need to state one definition and two more hypotheses. The first hypothesis is a bound for the largest eigenvalues and the other is an error estimate for the correction produced by the (k-1)-level iteration. Finally, we prove a lemma describing what effect the smoothing iterations have on the error.

<u>Definition 2.5</u>: The error on level k at the i^{th} stage of Algorithm 2.4 will be denoted by

$$e_i = u_k - z_i.$$

The first hypothesis states what form the bound for the maximum eigenvalue is assumed to have.

<u>Hypothesis 2.6</u> (Maximum Eigenvalue): There exist positive constants δ and C_2, each independent of j, such that

$$\Lambda_j \le C_2 N_j^{2\delta}, \; 1 \le j \le k.$$

The use of Λ_k in the smoothing iteration (2.7) may be replaced by any upper bound satisfying Hypothesis 2.6.

The last hypothesis is a norm estimate:

<u>Hypothesis 2.7</u> (Approximating Error Estimate): For some α with $0 < \alpha \le 1$, there exists a positive constant C_3 such that

$$|||E_k\bar{q}-e_n|||_{1-\alpha} \le C_3^{\alpha} N_k^{-2\alpha\delta} \; |||e_n|||_{1+\alpha}.$$

For problems derived from elliptic boundary-value problems, the value of α depends on the spatial domain.

The following lemma is used in the convergence proof to analyze the effect of the smoothing on elements in the solution spaces.

<u>Lemma 2.8</u>: Let $n > 0$ be any integer and $z_0 \in M_k$. Then the smoothing iteration (2.7) is a contraction operator:

$$|||e_n||| \le |||e_0||| . \qquad (2.10)$$

Further, for every fixed $0 \le \omega \le 2$ and $0 < \alpha \le 1$,

$$|||e_n|||_{\omega} \le C_2^{\alpha/2} N_k^{\alpha\delta} \; (2n + \alpha)^{-\alpha} \; |||e_0|||_{\omega-\alpha}. \qquad (2.11)$$

<u>Proof</u>: Let $N = N_k$ and $\Lambda = \Lambda_k$. From (2.7) we can deduce that

$$(e_i - e_{i-1}, v)_k = -\Lambda^{-1} a_k(e_{i-1}, v), \text{ for all } v \in M_k,\ i \geq 1. \quad (2.12)$$

We can expand e_0 in terms of the eigenfunctions:

$$e_0 = \sum_{i=1}^{N} \beta_i \xi_i.$$

Using (2.12) we can show that

$$e_n = \sum_{i=1}^{N} \beta_i (1 - \lambda_i/\Lambda)^n \xi_i. \quad (2.13)$$

Since $(1 - \lambda_i/\Lambda)^n \leq 1$, we have that

$$|||e_n||| \leq |||e_0|||.$$

The proof of (2.11) uses (2.10) and the Maximum Eigenvalue Hypothesis:

$$\begin{aligned}
|||e_n|||_\omega^2 &= \sum_{i=1}^{N} \beta_i^2 \lambda_i^\omega (1 - \lambda_i/\Lambda)^{2n} \\
&= \Lambda^\alpha \sum_{i=1}^{N} \beta_i^2 \lambda_i^{\omega-\alpha} (\lambda_i/\Lambda)^\alpha (1 - \lambda_i/\Lambda)^{2n} \\
&\leq \Lambda^\alpha \max_{x \in [0,1]} |x^\alpha (1 - x)^{2n}| \sum_{i=1}^{N} \beta_i^2 \lambda_i^{\omega-\alpha} \\
&\leq \Lambda^\alpha (2n + \alpha)^{-\alpha} |||e_0|||_{\omega-\alpha}^2 \\
&\leq C_2^\alpha N^{2\alpha\delta} (2n + \alpha)^{-\alpha} |||e_0|||_{\omega-\alpha}^2.
\end{aligned}$$

Taking the square root of both sides of this inequality completes the

proof.

QED

The convergence of Algorithm 2.4 is established in the Theorem 2.9. In essence, this result says that the error on level k can be reduced by any positive constant less than one provided the correction recursion problem on level k-1 can be solved sufficiently accurately.

<u>Theorem 2.9</u> (Convergence of Algorithm 2.4): Assume that Hypotheses 2.3, 2.6, and 2.7. Let $p > 1$ be any fixed integer. For any constant $0 < \gamma < 1$ there exist a nonnegative integer I which depends only on p and γ, such that

$$|||e_{m+n+1}||| \leq \gamma |||e_0|||, \text{ for all } m+n \geq I. \quad (2.14)$$

<u>Proof</u>: This proof is motivated by the work of Bank and Dupont [9]. The basic idea of this proof is to show that the smoothing iteration (2.7) reduces the oscillatory components of the error (corresponding to the larger eigenvalues) while the correction recursion iteration (2.8) reduces the smoother components of the error. The proof is by induction on the index k of the space. Assume the result is true for 1, 2, ... , k-1. We now prove the result for M_k.

By Lemma 2.8 (with $\omega = 1 + \alpha$),

$$|||e_n||| \leq |||e_0|||$$

and

$$|||e_n|||_{1+\alpha} \le C_2^{\alpha/2} N^{\alpha\delta} (2n + \alpha)^{-\alpha} |||e_0||| . \qquad (2.15)$$

We can estimate the effect of the correction recursion iteration (2.8). Using (2.8) and the Energy Norm Consistency Hypothesis, we have for all $v \in M_{k-1}$,

$$a_{k-1}(\bar{q},v) = C_1^{-1} a_k(e_n,E_k v)$$

or

$$a_k(E_k\bar{q}-e_n,E_k v) = 0. \qquad (2.16)$$

This shows that the exact solution $\bar{q}$ of the (k-1)-level correction recursion problem corrects exactly all components in z_n which belong to M_{k-1}. Take $v = \bar{q}$ in (2.16). Then

$$a_k(E_k\bar{q}-e_n,E_k\bar{q}) = 0 \Rightarrow a_k(E_k\bar{q},E_k\bar{q}) = a_k(e_n,E_k\bar{q})$$

$$\Rightarrow |||E_k\bar{q}||| \le |||e_n||| \le |||e_0||| .$$

Using this, the Energy Norm Consistency Hypothesis, and the induction hypothesis gives us

$$|||E_k(q-\bar{q})||| = C_1^{-1/2} |||q-\bar{q}||| \le C_1^{-1/2} \gamma^p |||\bar{q}||| = \gamma^p |||E_k\bar{q}|||$$

$$\le \gamma^p |||e_0||| . \qquad (2.17)$$

We are now ready to estimate $|||e_{m+n+1}|||$. We can define

$$e_{n+1} = E_k q-e_n$$

$$= (E_k\bar{q}-e_n) + E_k(q-\bar{q}).$$

If S^m reflects the effect of m smoothing iterations on any $v \in M_k$, then

$$e_{m+n+1} = S^m e_{n+1}.$$

Define $C_4 = C_2^\alpha C_3^\alpha$. Using Lemma 2.8 (with $\omega = 1$), (2.15), (2.17), and the Approximating Error Estimate Hypothesis yields

$$|||e_{m+n+1}||| \le |||S^m(E_k\bar{q}-e_n)||| + |||S^m E_k(q-\bar{q})|||$$

$$\le C_2^{\alpha/2} (2m + \alpha)^{-\alpha/2} N^{\alpha\delta} |||E_k\bar{q}-e_n|||_{1-\alpha} + |||E_k(q-\bar{q})|||$$

$$\le C_2^{\alpha/2} C_3^{\alpha} (2m + \alpha)^{-\alpha/2} N^{-\alpha\delta} |||e_n|||_{1+\alpha} + \gamma^p |||e_0|||$$

$$\le [C_4(2m + \alpha)^{-\alpha/2}(2n + \alpha)^{-\alpha/2} + \gamma^p] |||e_0||| . \qquad (2.18)$$

Choose I such that

$$C_4(2m + \alpha)^{-\alpha/2}(2n + \alpha)^{-\alpha/2} \le \gamma - \gamma^p, \text{ for all } m+n \ge I.$$

Then

$$|||e_{m+n+1}||| \le \gamma |||e_0||| .$$

Clearly, γ and I can be chosen independent of the N_j.

QED

At first glance, it appears that choosing m = n in (2.18) would be better than choosing either n = 0, m > 0 or n > 0, m = 0. However, special case proofs show that this is not necessarily true. Assume that $n > 0$, $m = 0$. By modifying the proof of Theorem 2.9, we can show that

$$|||e_{n+1}||| \leq [C_4^{1/2}(2n + \alpha)^{-\alpha/2} + \gamma^p] \; |||e_0||| .$$

For n large, this bound for the error reduction looks like

$$C_4^{1/2} \; (2n)^{-\alpha/2} \leq \varepsilon \equiv \gamma - \gamma^p .$$

So

$$n \geq \frac{1}{2} \, C_4^{1/\alpha} \, \varepsilon^{-2/\alpha}$$

is required to reduce the error by a factor of ε. If $\tilde{n} = \tilde{m}$, the error reduction (see (2.18)) looks like

$$C_4 \, \tilde{n}^{-\alpha} \leq \varepsilon .$$

Once again,

$$\tilde{n} \geq \frac{1}{2} \, C_4^{1/\alpha} \, \varepsilon^{-1/\alpha} = 2\varepsilon^{1/\alpha} \cdot \frac{1}{2} \, C_4^{1/\alpha} \, \varepsilon^{-2/\alpha} = 2\varepsilon^{1/\alpha} n$$

is required to reduce the error by a factor of ε. The amount of work, which depends on n and $2\tilde{n}$, depends on the relative sizes of $2^{\alpha+1}\varepsilon$ and one. Thus, we have shown

<u>Theorem 2.10</u>: Assume Theorem 2.9 holds. Define $\varepsilon \equiv \gamma - \gamma^p$. Then choosing $n > 0$, $m = 0$ in (2.18) is better than choosing $\tilde{n} = \tilde{m} > 0$ only when $2^{\alpha+1}\varepsilon > 1$. Alternately, choosing $\tilde{n} = \tilde{m} > 0$ in (2.18) is better only when $2^{\alpha+1}\varepsilon < 1$.

The practical significance of this is that if $\gamma - \gamma^p$ is large, it is more efficient to do the smoothing at once, rather than splitting it around the correction recursion. Similar analysis holds for the case of $n = 0$, $m > 0$.

We now analyze the cost of one iteration of Algorithm 2.4. Let F(N) be the cost of reducing the error by a factor of γ for a problem with N unknowns. We assume that the cost of the smoothing iterations (2.7) (or an iteration with similar properties) on level k can be bounded by $C_5(m+n)N_k \equiv C_6 N_k$, where C_5 is independent of k. The cost of the correction recursion (2.8) is $pF(N_{k-1})$. Thus,

$$F(N_k) \sim pF(N_{k-1}) + C_6 N_k . \qquad (2.19)$$

Since $N_k \sim \sigma N_{k-1}$ (see (2.2)), the solution of (2.19) is (asymptotically in k)

$$F(N) \leq \begin{cases} C_7 N, & 2 \leq p < \sigma \\ C_7 N \log N, & p = \sigma \\ C_7 N^{\log p}, & p > \sigma, \end{cases} \qquad (2.20)$$

where the logarithms are base σ [2].

Choosing $2 \leq p < \sigma$ leads to an optimal order algorithm, in the sense that the error can be reduce by a fixed factor of γ each iteration with work proportional to the number of unknowns. However, we may want to reduce the initial error by a factor of $N^{-q\delta}$ for some fixed q. The obvious implementation would then require F(N)logN operations. We assume the solutions u_j of Problem 2.2 satisfy

$$|||p_j u-u_j||| \leq KN_j^{-q\delta}, \ j \geq 1,$$

where K is a constant independent of N_j. Denote by $\tilde{u}_j$ the computed solution of the j-level scheme (Algorithm 2.4). To avoid the extra logN factor, we use $E_j\tilde{u}_{j-1}$ as the initial guess to $\tilde{u}_j$, $j > 1$, and prove that the initial error is small. Approximate solutions $\tilde{u}_j$ of finite-dimensional Problem 2.2 are generated using the following:

<u>Algorithm</u> <u>2.11</u>: Given an integer $j > 0$ and $\{M_i, a_i(\cdot,\cdot), f_i(\cdot)\}_{i=1}^{j}$, we want to approximate $u_j \in M_j$, where $a_j(u_j,v) = f_j(v)$ for all $v \in M_j$.

(a) If $j = 1$, then solve Problem 2.2 directly.

(b) If $j > 1$, then starting from an initial guess $z_0 = E_j\tilde{u}_{j-1}$, apply r iterations of the j-level scheme (Algorithm 2.4) to Problem 2.2 to obtain $\tilde{u}_j$.

This algorithm actually has four parameters. The first parameter, r, determines the number of iterations of the j-level scheme to use. The j-level scheme itself has three parameters: m and n, the number of smoothing iterations and p, the number of correction recursion iterations.

Figure 2-3 contains a three-level example of one iteration ($r = 1$) of Algorithm 2.11. For the correction recursion problems, $p = 2$. Unlike Algorithm 2.4, computation begins with the smallest space and winds its way "down" to the largest space. It is worth noting that when $n = 0$ and $z_0 = 0$ in Figure 2-2, the two algorithms become much more similar. In this case, the first recursion correction problem

Figure 2-3: Three-Level Example of Algorithm 2.11

One iteration ($r = 1$) on level three, $p = 2$

```
   Level

    1  ds   ds       ds   ds
         \ /  \      /  \ /  \
    2     n    m  n   n+m    m
                \ /            \
    3            n              m

     ds = direct solve
    n,m = number of smoothing iterations
```

(see (2.8)) in Figure 2-2 has $\bar{f}_k(v) = f_k(v)$.

The convergence properties of Algorithm 2.11 are stated in the Theorem 2.12.

<u>Theorem</u> <u>2.12</u>: Assume that Hypotheses 2.3, 2.6, and 2.7 hold. Let $r \geq 1$ be any fixed integer. Suppose

(i) $|||p_j u-u_j||| \leq KN_j^{-q\delta}$, $j \geq 1$,

(ii) $|||p_j u-E_j p_{j-1}u||| \leq KN_j^{-q\delta}$, $j \geq 2$, and

(iii) $|||u_1-\tilde{u}_1||| \leq KN_1^{-q\delta}$.

Then

(a) For any integer $p > 1$, there exists a nonnegative integer I such that for $j \geq 1$,

$$|||u_j-\tilde{u}_j||| \leq KN_j^{-q\delta}, \text{ for all } m+n \geq I.$$

(b) $|||p_ju-\tilde{u}_j||| \leq 2KN_j^{-q\delta}$

(c) the cost of computing $\tilde{u}_j$ is bounded by $C_8F(N_j)$, where $C_8 = \sigma r/(\sigma-1)$ is independent of j.

Proof: (a) The proof is by induction on the index j of the space. Assume the result is true for 1, 2, ... , j-1. We now prove the result for M_j. Define $\tilde{e}_j = |||u_j-\tilde{u}_j|||$ for $j \geq 1$. After r iterations of the j-level scheme,

$$\begin{aligned}\tilde{e}_j &\leq \gamma^r\ |||u_j-E_j\tilde{u}_{j-1}|||\\ &\leq \gamma^r\ \{\ |||u_j-p_ju||| + |||p_ju-E_jp_{j-1}u||| +\\ &\qquad |||E_j(p_{j-1}u-u_{j-1})||| + |||E_j(u_{j-1}-\tilde{u}_{j-1})|||\ \}\\ &\leq \gamma^r\ \{\ K(2 + C_1^{1/2} + C_1^{-1/2}\sigma^{q\delta})N_j^{-q\delta}.\end{aligned}$$

Choose $\gamma \leq (2 + C_1^{1/2} + C_1^{-1/2}\sigma^{q\delta})^{-1/r}$ and I according to Theorem 2.9. Hence,

$$\tilde{e}_j \leq KN_j^{-q\delta}.$$

(b) This is a simple consequence of the triangle inequality:

$$\begin{aligned}|||p_ju-\tilde{u}_j||| &\leq |||p_ju-u_j||| + |||u_j-\tilde{u}_j|||\\ &\leq 2KN_j^{-q\delta}.\end{aligned}$$

(c) The cost of computing $\tilde{u}_j$ is bounded by

$$F(N_1) + r\sum_{i=2}^{j} F(N_i) \leq C_7\ F(N_j)\ \sigma r/(\sigma-1) = C_8\ F(N_j).$$

QED

It is worth pointing out that r is independent of j. This tells us that Algorithm 2.11 is optimal order whenever $2 \leq p < \sigma$.

In conclusion, we have shown that solutions to problems like Problem 2.1 can be approximated in finite-dimensional spaces using an optimal order procedure provided that $\sigma > 2$. When $\sigma = 2$, the operation count is $O(N_k\log N_k)$. In the next three chapters we verify that Hypotheses 2.3, 2.6, and 2.7 hold for particular discretizations of several elliptic boundary-value problems.

CHAPTER 3

Finite-Element Analysis

In this chapter, we discuss the iterative solution of large sparse linear systems which arise in connection with finite-element procedures for solving self-adjoint elliptic boundary-value problems. We show that the algorithms and theorems of Chapter 2 apply to this case. We will see that this is an optimal order result.

Our model is the Neumann problem

$$\begin{cases} -\nabla\cdot(P\nabla u) + Su = f \text{ in } \Omega \\ u_n = 0 \text{ on } \partial\Omega, \end{cases} \tag{3.1}$$

where Ω is a polygonal domain in R^2. We assume that $P \in C^1(\bar{\Omega})$, $S \in C(\bar{\Omega})$, and that there exist positive constants $\underline{p}$, $\bar{p}$, $\underline{s}$, and $\bar{s}$ such that

$$\underline{p} \le P(x) \le \bar{p},\ \underline{s} \le S(x) \le \bar{s}, \text{ for all } x\in\bar{\Omega}.$$

Most of our arguments apply to the Dirichlet problem

$$\begin{cases} -\nabla\cdot(P\nabla u) + Su = f \text{ in } \Omega \\ u = 0 \text{ on } \partial\Omega \end{cases} \tag{3.2}$$

with only minor modifications. We comment on the the extensions as they arise.

We seek a weak form solution of (3.1): find $u \in H^1(\Omega)$ such that

$$a(u,v) = (f,v) \quad \text{for all } v \in H^1(\Omega), \tag{3.3}$$

where

$$\begin{aligned} a(u,v) &= \int_\Omega P\nabla u\cdot\nabla v + Suv\, dx \quad \text{and} \\ (f,v) &= \int_\Omega fv\, dx. \end{aligned} \tag{3.4}$$

Then there exists a unique weak solution $u \in H^1(\Omega)$ for all $f \in L^2(\Omega)$ (see Ciarlet [16]). The spaces H^s, for s a positive integer, will be the usual Sobolev spaces equipped with norms

$$\|u\|_s^2 = \sum_{|\beta|\le s} \|D^\beta u\|_s^2 = \sum_{|\beta|\le s} (D^\beta u, D^\beta u).$$

The spaces H^s for s positive and non-integral will be defined by interpolation (see Agmon [1] or Lions and Magenes [36]). For s negative, H^s will be defined as the dual of H^{-s}. The bilinear form $a(\cdot,\cdot)$ induces the energy norm $|||u|||^2 = a(u,u)$.

A modest amount of elliptic regularity for the solution of (3.3) is required.

__Hypothesis 3.1__ (Regularity): We assume there exists a constant $0 < \alpha \leq 1$, such that for all $f \in H^{\alpha-1}$ there exists a unique solution $u \in H^{1+\alpha}$ of (3.3) and

$$\|u\|_{1+\alpha} \leq C(P,S,\Omega)\ \|f\|_{\alpha-1}.$$

For a complete discussion of what values of α correspond to specific domains Ω, see Kellog [34] and Babuska and Aziz [7].

We now consider a finite-element approximation of (3.3). Let T_j, $j \geq 1$, be a nested sequence of triangulations of Ω. Take T_1 to be a fixed triangulation. For $T \in T_1$, denote the diameter of T by h_T, and let $h_T \cdot d_T$ denote the diameter of the inscribing circle for T. Define

$$h_1 = \max_{T \in T_1} h_T,$$

$$\delta_0 = \min_{T \in T_1} d_T, \quad \text{and}$$

$$\delta_1 = \min_{T \in T_1} h_T/h_1.$$

The constant δ_0 is a measure of the shape regularity of the triangles in T_1 and δ_1 is a measure of their uniformity. We construct T_j, $j > 1$, inductively: divide every $T \in T_{j-1}$ into μ^2 congruent triangles, where μ is independent of j. When $\mu = 2$, this means we construct four triangles in T_j by pairwise connecting the midpoints of the edges. Each triangulation T_j will have shape regularity and uniformity constants δ_0 and δ_1 and will have $h_j = \max_{T \in T_j} h_T = \mu^{1-j} h_1$.

With each triangulation T_j, we associate the N_j-dimensional space M_j of C^0-piecewise linear polynomials. Following (2.2), we know that

$$N_{j+1} \sim \sigma N_j, \tag{3.5}$$

where $\sigma = \mu^2$ asymptotically. Since the triangulations are nested, we have that M_j is a subset of M_{j+1}, $j \geq 1$. The spaces M_j satisfy the following standard approximation property [11, 12, 33, 44]: if $u \in H^s$, $1 \leq s \leq 1+\alpha$, then there exists a $u_j \in M_j$ such that

$$\|u-u_j\|_0 + h_j\|u-u_j\|_1 \leq C(\delta_0,\delta_1,\Omega)\ h_j^s\ \|u\|_s. \tag{3.6}$$

We briefly remark on the Dirichlet model problem (3.2). The definition of T_j, $j \geq 1$, remains the same. Let M_j be a subset of H_0^1 and let it be the space of C^0-piecewise linear polynomial associated with T_j satisfying the Dirichlet boundary conditions. Then M_j is a subset of M_{j+1}, $j \geq 1$, as before. With this modification, the conclusions of the lemmas and theorems of this chapter will remain valid.

Using the notation of Chapter 2, we define for each space M_j, $j \geq 1$,

$$a_j(u,v) = a(u,v),$$

$$f_j(v) = (f,v), \quad \text{and}$$

$$(u,v)_j = (u,v) \text{ for all } u,v \in M_j.$$

The interpolation and projection operators, E_j, p_j, and i_j (see (2.3) and (2.4), are the natural projections and injections.

Associated with M_j are eigenvalues $\lambda_i^{(j)}$, eigenfunctions $\xi_i^{(j)}$, and a maximum eigenvalue Λ_j of $a(\cdot,\cdot)$ fulfilling the requirements of (2.6), where $1 \le i \le N_j$. A simple homogeneity argument shows that

$$\Lambda_j \le C(P,S,\delta_0,\delta_1,\Omega)h_j^{-2} \tag{3.7}$$

(see Strang and Fix [48]). For $-2 \le s \le 2$, we define discrete norms

$$|||v|||_s^2 = \sum_{i=1}^{N} c_i^2 (\lambda_i^{(j)})^s, \quad \text{for} \quad v = \sum_{i=1}^{N} c_i \xi_i^{(j)}, \tag{3.8}$$

where we have suppressed the j subscript on the norm. Note that $|||v|||_1 = |||v|||$ and $|||v|||_0$ is comparable to $\|v\|_0$. In fact, the proof of the following norm equivalence is almost identical to Lemma 1 in Bank and Dupont [9]:

<u>Lemma 3.2</u>: There exists a constant $C = C(P,S,\Omega,\delta_0,\delta_1,\beta)$ such that for $0 \le s \le 1$,

$$C^{-1}\|v\|_s \le |||v|||_s \le C\|v\|_s.$$

In order to establish the convergence of Algorithms 2.4 and 2.11 for the finite-element case, we must verify Hypotheses 2.3, 2.6, and 2.7. It is immediate that (the Energy Norm Consistency) Hypothesis 2.3 holds. Using (3.7) and the fact that $N_j \sim Ch_j^{-2}$ proves that (the Maximum Eigenvalue) Hypothesis 2.6 holds.

A duality argument is used to verify (the Approximating Error

Estimate) Hypothesis 2.7. When α is an integer, this is a standard result [16, 48].

<u>Lemma 3.3</u>: Let α be defined by the Regularity Hypothesis. Then

$$|||\bar{q}-e_n|||_{1-\alpha} \le C\mu^{2\alpha} h_k^{2\alpha} |||e_n|||_{1+\alpha}.$$

<u>Proof</u>: By duality and Lemma 3.2,

$$|||\bar{q}-e_n|||_{1-\alpha} \le C\|\bar{q}-e_n\|_{1-\alpha}$$

$$= C \sup_{\rho\varepsilon H^{\alpha-1}} \frac{(\rho,\bar{q}-e_n)}{\|\rho\|_{\alpha-1}} \tag{3.9}$$

For $\rho \varepsilon H^{\alpha-1}$, let $\eta \varepsilon H^{\alpha+1}$ be defined by

$$a(\eta,v) = (\rho,v) \text{ for all } v \varepsilon H^1.$$

Taking $v = \bar{q}-e_n$ gives us

$$(\rho,\bar{q}-e_n) = a(\eta,\bar{q}-e_n)$$

$$= a(\eta-\omega,\bar{q}-e_n), \text{ for any } \omega \varepsilon M_{k-1}.$$

By the Regularity Hypothesis and (3.6),

$$(\rho,\bar{q}-e_n) \le Ch_{k-1}^{\alpha} \|\eta\|_{\alpha+1} |||\bar{q}-e_n|||$$

$$\le C\mu^{\alpha} h_k^{\alpha} \|\rho\|_{\alpha-1} |||\bar{q}-e_n||| .$$

Combining this with (3.9) yields

$$|||\bar{q}-e_n|||_{1-\alpha} \le C\mu^{\alpha} h_k^{\alpha} |||\bar{q}-e_n||| . \tag{3.10}$$

However,

$$|||\bar{q}-e_n|||^2 = a(\bar{q}-e_n,\bar{q}-e_n)$$

$$= - a(\bar{q}-e_n,e_n) \qquad (3.11)$$

$$\le |||\bar{q}-e_n|||_{1-\alpha}|||e_n|||_{1+\alpha}.$$

Substituting (3.10) into the right-hand side of (3.11) gives us

$$|||\bar{q}-e_n||| \le C\mu^{\alpha} h_k^{\alpha} |||e_n|||_{1+\alpha},$$

which we substitute back into (3.10) to complete the proof.

QED

This proves that Algorithm 2.4 converges at the rates specified by Theorem 2.9 for the finite-element case of this chapter.

One of the advantages of finite-element methods is that the theory of Chapter 2 can be applied using a variety of norms. As an example, we prove a special case of Theorem 2.9 for the L^2 norm. It is similar to the results of Nicolaides [38] for the l^2 norm and is the analogue of Corollary 1 of Bank and Dupont [9]. To get L^2 results, we assume that the solution u has H^2 regularity, i.e., $\alpha = 1$. This assumption requires that Ω be convex [26, 48].

<u>Theorem 3.4</u>: Assume the Regularity Hypothesis holds for $\alpha = 1$. Let $p > 1$ be any fixed integer. For any constant $0 < \gamma < 1$ there exists a nonnegative integer I, which depends only on p and γ, such that

$$\|e_{m+n+1}\|_0 \le \gamma\|e_0\|_0 \quad \text{for all } m+n \ge I.$$

<u>Proof</u>: The proof is by induction on the index k of the space. Assume the result is true for 1, 2, ... , k-1. We now prove the result for M_k. From (2.13) it is immediate that

$$\|e_n\|_0 \le \|e_0\|_0. \qquad (3.12)$$

Using an argument similar to the proof of Lemma 2.8 shows that

$$\|e_n\|_2 \le C\mu^{-2}h_k^{-2}(2n+1)^{-1}\|e_0\|_0. \qquad (3.13)$$

Lemmas 3.2 and 3.3 with $\alpha = 1$ yield

$$\|\bar{q}-e_n\|_0 \le C\mu^2 h_k^2 \|e_n\|_2. \qquad (3.14)$$

The analogue of (2.17) is derived using the induction hypothesis and (3.12) - (3.14):

$$\|q-\bar{q}\|_0 \le \gamma^p\|\bar{q}\|_0$$

$$\le \gamma^p\{\|\bar{q}-e_n\|_0 + \|e_n\|_0\}$$

$$\le \gamma^p\{C\mu^2h_k^2\|e_n\|_2 + \|e_0\|_0\}$$

$$\le \gamma^p\{C(2n+1)^{-1} + 1\}\|e_0\|_0$$

$$\le C\gamma^p\|e_0\|_0.$$

Using an argument similar to (2.18) gives us

$$\|e_{m+n+1}\|_0 \le C\{(2m+1)^{-1}(2n+1)^{-1} + \gamma^p\}\|e_0\|_0$$

and Theorem 3.4 follows.

QED

It is useful to associate with M_j a symmetric, positive definite bilinear form $b_j(\cdot,\cdot)$, which is comparable to the L^2 inner product. We assume there exists a constant β, independent of h_j, such that

$$0 < \beta^{-1} \leq \frac{b_j(u,v)}{(v,v)} \leq \beta \text{ for all } v \in M_j,\ v \neq 0. \qquad (3.15)$$

For $b_j(\cdot,\cdot)$, we define generalized eigenvalues $\tilde{\lambda}_i$ and eigenfunctions $\tilde{\xi}_i$, $1 \leq i \leq N_j$, by

$$a(v,\tilde{\xi}_i) = \tilde{\lambda}_i\, b_j(v,\tilde{\xi}_i) \text{ for all } v \in M_j,$$

$$0 < \tilde{\lambda}_1 \leq \tilde{\lambda}_2 \leq \dots \leq \tilde{\lambda}_{N_j} = \tilde{\Lambda}_j,$$

$$b_j(\tilde{\xi}_i,\tilde{\xi}_k) = \delta_{ik}, \quad \text{and}$$

$$a(\tilde{\xi}_i,\tilde{\xi}_k) = \tilde{\lambda}_i\delta_{ik},\ 1 \leq k \leq N_j.$$

It then follows that

$$\tilde{\Lambda}_j = \max_{\substack{v\in M_j \\ v\neq 0}} \frac{a(v,v)}{b_k(v,v)} \leq \beta \max_{\substack{v\in M_j \\ v\neq 0}} \frac{a(v,v)}{(v,v)} \leq C(P,S,\Omega,\delta_0,\delta_1,\beta)h_j^{-2}.$$

For $-2 \leq s \leq 2$, define $|||v|||_{s,b}^2$ similarly to $|||v|||_s^2$ (see (3.8)). Note that $|||v|||_{1,b} = |||v|||$ and $|||v|||_{0,b}$ is comparable to $\|v\|_0$. In fact, a norm equivalence similar to 3.2 can be proved with $|||v|||_{s,b}$ substituted for $|||v|||_s$.

The smoothing iteration (2.7),

$$(z_i - z_{i-1},v)_k = \Lambda_k^{-1}[f_k(v) - a_k(z_{i-1},v)], \text{ for all } v \in M_k,$$

requires the solution of a linear system involving the mass matrix at every step. In practice this is too expensive. We can replace (2.7) with the smoothing iteration

$$b_k(z_i - z_{i-1},v) = \tilde{\Lambda}_k^{-1}[(f,v) - a(z_{i-1},v)], \text{ for all } v \in M_k. \qquad (3.16)$$

There are numerous choices for $b_j(\cdot,\cdot)$, $j \geq 1$. When the standard nodal basis is used, (3.15) is satisfied by $b_j(\cdot,\cdot)$ corresponding to the diagonal of the mass matrix. This allows smoothing by an under-relaxed Jacobi scheme.

The convergence of the k-level scheme (Algorithm 2.4) is summarized by the following result. Its proof is almost identical to the ones for Theorems 2.9 and 3.4.

<u>Theorem 3.5</u>: Assume the Regularity Hypothesis holds. Define the k-level scheme (Algorithm 2.4) using (3.16) instead of (2.7). Let $\|\cdot\|$ denote either the energy or L^2 norm (and be fixed). Let $p > 1$ be any fixed integer. For any constant $0 < \gamma < 1$ there exists a nonnegative integer I, which depends only on p and γ, such that

$$\|e_{m+n+1}\| \leq \gamma\|e_0\| \text{ for all } m+n \geq I.$$

We conclude this chapter by noting that for the finite-element method of this chapter, Algorithm 2.11 converges at the rate specified by Theorem 2.12 for either the energy or L^2 norm. Let $\|\cdot\|$ denote either of these norms. Then Theorem 2.12 can be rewritten as follows:

<u>Theorem</u> <u>3.6</u>: Assume the Regularity Hypothesis holds. Let $r \geq 1$ be any fixed integer. Suppose

(i) $\|u-u_j\| \leq Kh_j^q$, $j \geq 1$, and

(ii) $\|u_1-\tilde{u}_1\| \leq Kh_1^q$.

Then for any integer $p > 1$, there exists a nonnegative integer I such that for $j \geq 1$,

$$\|u_j-\tilde{u}_j\| \leq Kh_j^q \text{ for all } m+n \geq I.$$

Moreover,

$$\|u-\tilde{u}_j\| \leq 2Kh_j^q,$$

and the cost of computing $\tilde{u}_j$ is bounded by $C_8F(N_j)$, where $C_8 = \sigma r/(\sigma-1)$ is independent of j and σ is defined by (3.5).

It is worth pointing out that r is independent of j. This implies that Algorithm 2.11 is optimal order when $2 \leq p < \sigma$.

CHAPTER 4
One-Dimensional Finite-Difference Analysis

4.1 Introduction

In this chapter, we discuss the iterative solution of large sparse linear systems which arise in connection with finite-difference approximations to the model problem:

$$\begin{cases} -(Pu_x)_x + Su = f \text{ in } \Omega \equiv (0,1) \\ u(0) = u(1) = 0. \end{cases} \tag{4.1}$$

We assume that $f \in L^2$ and there exist positive constants $\underline{p}$, $\bar{p}$, $\underline{s}$, and $\bar{s}$ such that

$$\underline{p} \leq P(x) \leq \bar{p}, \quad \underline{s} \leq S(x) \leq \bar{s}, \text{ for all } x \in \bar{\Omega}. \tag{4.2}$$

We seek a weak form solution of (4.1): find $u \in H_0^1(\Omega)$ such that

$$a(u,v) = (f,v) \text{ for all } v \in H,$$

where

$$a(u,v) \equiv \int_\Omega Pu_xv_x + Suv\,dx \quad \text{and}$$

$$(f,v) \equiv \int_\Omega fv\,dx.$$

Then there exists a unique weak solution $u \in H(\Omega)$ for all $f \in L^2(\Omega)$ (see Ciarlet [16]). The theory in this chapter restricts our attention to problems where the solution u lies in

$$H = H_0^1 \cap H^2.$$

The bilinear form $a(\cdot,\cdot)$ induces the energy norm $|||u|||^2 = a(u,u)$.

In practice, finite-difference approximations to (4.1) are solved directly. In Section 4.2, we show that the algorithms and theorems of Chapter 2 apply to this case, though the algorithms are not optimal order. We analyze this problem in two parts. First, the case of $P = 1$ and $S = 0$ in (4.1) is considered. Then we analyze the general case. In Section 4.3, we consider the case when only two grids are used. We present a summary of sharper results that are be proved in Douglas [19]. We first investigate how the order of interpolation affects the rate of convergence. We restrict our attention to the cases of piecewise-linear and piecewise-cubic interpolation between grids. This can be used to optimize the number of smoothing iterations and correction recursions to minimize the amount of work required.

4.2 K Levels

In this section, we assume that the number of grids is arbitrarily large. We prove that the theorems and algorithms of Chapter 2 apply to a particular discretization of (4.1). Most of this section contains analysis of a simple problem. At the end of this section, we extend the analysis to the elliptic problem (4.1).

Let $k > 1$ and $N_0 > 0$ be fixed integers. We define $N_j = 2N_{j-1} + 1$,

$j \geq 1$. Following (2.2), we know that $\sigma = 2$. We define k uniform grids Ω^j, $1 \leq j \leq k$, by

$$h_k = (N_k + 1)^{-1},$$

$$h_j = 2^{k-j}h_k, \quad \text{and}$$

$$\Omega^j = \{ ih_j \mid 1 \leq i \leq N_j \}.$$

For the moment, we assume that $P = 1$ and $S = 0$ in (4.1). The general case will be analyzed at the end of the chapter. We discretize (4.1) by approximating the second derivatives by central differences to obtain a system of linear equations

$$A_k u_k = h_k f_k, \tag{4.3}$$

where u_k and f_k approximate the solution u and the right-hand side f on Ω^k. The $N_k \times N_k$ matrix A_k is given by

$$A_k = h_k^{-1} \begin{vmatrix} 2 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & & \cdot & & & \\ & & & & \cdot & & \\ & & & & \cdot & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 2 & -1 \\ & & & & & -1 & 2 \end{vmatrix} \tag{4.4}$$

(see Varga [50]). Factoring and solving this algebraic system directly requires only $5N_k$ multiplications and $3N_k$ additions [25]. We will show that the multi-grid algorithms of Chapter 2 require $O(N_k \log N_k)$ operations to solve this problem to accuracy comparable to the discretization error, i.e., $O(N_k^{-2})$.

We associate a <u>solution space</u> M_j, $1 \leq j \leq k$, with each grid. Let $N = N_k$. Then

$$M_k = \{ v \mid v \in R^N \} \quad \text{and}$$

$$M_{k-1} = \{ v \mid v \in M_k \text{ and } v = \langle 0, v_1, 0, v_2, 0, \ldots, 0, v_{N_{k-1}}, 0 \rangle \}. \tag{4.5}$$

The spaces $M_{k-2}, \ldots, M_1$ are defined recursively.

We define interpolation operators between adjacent solution spaces

E: $M_{j-1} \to M_j$ and

R: $M_j \to M_{j-1}$

by means of the matrices

$$E = \left| \begin{array}{cccccccccccccccc}
0 & b-a & 0 & a & & & & & & & & & & & & \\
0 & 1 & & & & & & & & & & & & & & \\
0 & b & 0 & b & 0 & a & & & & & & & & & & \\
\cdot & & & 1 & & & & & & & & & & & & \\
\cdot & a & 0 & b & 0 & b & 0 & a & & & & & & & & \\
\cdot & & & & & 1 & & & & & & & & & & \\
 & & & a & 0 & b & 0 & b & 0 & a & & & & & & \\
 & & & & & & & \ddots & & & & & & & & \cdot \\
 & & & & & & & & & & & & & & & \cdot \\
 & & & & & & & & & & & & 1 & & & \cdot \\
 & & & & & & & & & & a & 0 & b & 0 & b & 0 \\
 & & & & & & & & & & & & & & 1 & 0 \\
 & & & & & & & & & & & & a & 0 & b-a & 0
\end{array} \right| \tag{4.6}$$

and

$$R = E^T.$$

In (4.6), we are taking advantage of two facts about the solution u_k:

it is zero at the boundaries and it can be continued outside the domain by reflection. We single out two choices of a and b for further study:

<u>Definition 4.1</u>: The <u>piecewise-linear interpolant</u>, E_2, is defined by substituting $a = 0$ and $b = 1/2$ in (4.6). The <u>piecewise-cubic interpolant</u>, E_4, is defined by substituting $a = -1/16$ and $b = 9/16$.

For the multi-grid analysis of Chapter 2, piecewise-linear interpolation will suffice. In Section 4.3, we will analyze the effect of linear and cubic interpolation on the order of convergence of the multi-grid algorithms of Chapter 2 when they are restricted to two levels.

Problems on coarser grids are defined using the interpolation matrices. Linear systems on coarser grids, A_j, $1 \leq j < k$, are defined recursively:

$$A_{k-j} = E_2^T A_{k-j+1} E_2.$$

Let $r \in R^N$ be a right-hand side for level k. Then the <u>(k-1)-level problem</u> is defined by

$$A_{k-1} \bar{q} = E^T r = f_{k-1}, \quad \bar{q} \in M_{k-1}, \tag{4.7}$$

where E is either the linear or cubic interpolant. For the analysis of Chapter 2, E in (4.7) is the linear interpolant. Half the rows and columns of A_{k-1} are zero. We can re-order the matrix so that the nonzero rows and columns are the first N_{k-1} rows and columns. Then A_{k-1} has a submatrix whose form is identical to A_k. The re-ordered solution space M_{k-1} has the form

$\{ v \in M_k \mid v = \langle v_1, v_2, \ldots, v_{N_{k-1}}, 0, \ldots, 0 \rangle \}$.

The (k-i)-level problems are defined similarly.

Before the theory of Chapter 2 can be applied, we must complete the definitions required for the triples used by Problems 2.1 and 2.2. For $j \geq 1$, a bilinear form $a_j(\cdot,\cdot)$, a linear functional $f_j(v)$, and an inner product $(\cdot,\cdot)_j$ are defined by

$$\begin{aligned} a_j(u,v) &= u^T A_j v \\ f_j(v) &= h_j f_j^T v, \text{ for all } u,v \in M_j, \quad \text{and} \qquad (4.8) \\ (u,v)_j &= h_j u^T v. \end{aligned}$$

The bilinear form $a_j(\cdot,\cdot)$ induces the energy norm

$$|||u|||^2 = a_j(u,u)$$

for all $u \in M_j$. Associated with each $a_j(\cdot,\cdot)$ are $N = N_j$ nonzero eigenvalues $\lambda_i^{(j)}$ and eigenvectors $\xi_i^{(j)}$ satisfying (2.6). Let Λ_j be the largest eigenvalue. For $-2 \leq s \leq 2$, discrete norms are defined by

$$|||v|||_s^2 = \sum_{i=1}^{N} \beta_i^2 (\lambda_i^{(j)})^s \quad \text{for} \quad v = \sum_{i=1}^{N} \beta_i \xi_i^{(j)}, \qquad (4.9)$$

where we have suppressed the j subscript on the norm. Note that $|||v|||_1 = |||v|||$ is the usual energy norm on level j. Hereafter, we drop the superscripts from the eigenvalues and eigenvectors.

In order to establish the convergence of Algorithms 2.4 and 2.11

for the finite-difference case of this chapter, we must verify Hypotheses 2.3, 2.6, and 2.7. We begin by showing that (the Energy Norm Consistency) Hypothesis 2.3 holds. It is derived using the definition of the linear systems A_{j-1}, $j > 1$.

<u>Lemma 4.2</u>: Let $j > 1$ be an integer. Then

$$a_j(E_2v, E_2w) = a_{j-1}(v,w), \text{ for all } v,w \in M_{j-1}.$$

<u>Proof</u>: For any $v,w \in M_{j-1}$,

$$\begin{aligned} a_j(E_2v,E_2w) &= (E_2v)^T A_j (E_2w) \\ &= v^T (E_2^T A_j E_2) w \\ &= v^T A_{j-1} w \\ &= a_{j-1}(v,w). \end{aligned}$$

QED

That (the Maximum Eigenvalue) Hypothesis 2.6 holds is a simple consequence of the explicit formula for the eigenvalues of A_j.

<u>Lemma 4.3</u>: $\Lambda_j \leq 4h_j^{-2}$.

<u>Proof</u>: The eigenvalues of $a_j(\cdot,\cdot)$ are

$$\lambda_i = 2h_j^{-2}(1 - \cos(i\pi h_j)), \ 1 \leq i \leq N_j,$$

from which the result is derived.

QED

In order to prove that (the Approximating Error Estimate) Hypothesis 2.7 holds, we must first prove two auxiliary norm estimates. These estimates rely on the fact that the interpolating matrices E_2 and E_4 can be block diagonalized using a sine transform. Let

$$Q = [\ Q_{\alpha\beta}\ ], \quad \text{where}$$
$$Q_{\alpha\beta} = \{\ 2h\ \}^{1/2} \sin(\alpha\pi\beta h). \tag{4.10}$$

The columns of Q are the eigenvectors of the matrix A_k. Note that Q is an orthogonal matrix. By direct computation, we can prove

<u>Proposition</u> <u>4.4</u>: Let E be defined by (4.6) and Q by (4.10). Let δ_{ij} be the Kronecker delta function. Then

$$\begin{aligned}[\ QEQ^{-1}\ ]_{ij} &= [\ QEQ\ ]_{ij} \\ &= \tfrac{1}{2}\{1 + 2[b\cos(i\pi h_j) + a\cos(3i\pi h_j)]\} \\ &\qquad \cdot (\delta_{ij} - \delta_{N+1-i,j}).\end{aligned}$$

For $i = 1, 2, \ldots, (N_j-1)/2$, define

$$x_i = \tfrac{1}{2}\{\ 1 + \cos(i\pi h_j)\ \} \quad \text{and}$$
$$y_i = \tfrac{1}{2}\{\ 1 - \cos(i\pi h_j)\ \}. \tag{4.11}$$

We can visualize QE_2Q^{-1} by

$$QE_2Q^{-1} = \begin{vmatrix} x_1 & & & & & -x_1 \\ & x_2 & & & -x_2 & \\ & & \ddots & \cdot & & \\ & & \cdot & \ddots & & \\ & -y_2 & & & y_2 & \\ -y_1 & & & & & y_1 \end{vmatrix}. \tag{4.12}$$

Similarly, we define $\bar{x}_i$ and $\bar{y}_i$ for QE_4Q^{-1}:

$$\bar{x}_i = \tfrac{1}{2}\{\ 1 + \tfrac{1}{8}[\ 9\cos(i\pi h_j) - \cos(3i\pi h_j)\ ]\ \} \quad \text{and}$$
$$\bar{y}_i = \tfrac{1}{2}\{\ 1 - \tfrac{1}{8}[\ 9\cos(i\pi h_j) - \cos(3i\pi h_j)\ ]\ \}. \tag{4.13}$$

Note that

$$x_i + y_i = 1 \quad \text{and}$$
$$\bar{x}_i + \bar{y}_i = 1 \tag{4.14}$$

for all $i = 1, 2, \ldots, (N_j-1)/2$.

We now prove the two auxiliary norm estimates.

<u>Lemma</u> <u>4.5</u>: Let $k > 1$ be an integer and $u \varepsilon M_k$. Then

$$\min_{v\varepsilon M_{k-1}} (u - E_2v)^T(u - E_2v) \leq \tfrac{1}{4}h_k^4\, u^T A_k^2\, u. \tag{4.15}$$

<u>Proof</u>: Let $N = N_k$. Since the odd numbered columns of E_2 are zero, the odd numbered elements of v make no difference. This important observation leads to the equality

$$\min_{v\varepsilon M_{k-1}} (u - E_2v)^T(u - E_2v) = \min_{v\varepsilon R^N} (u - E_2v)^T(u - E_2v). \tag{4.16}$$

It turns out that finding a $v \varepsilon R^N$ which minimizes the right-hand side of (4.16) is easier than finding a $v \varepsilon M_{k-1}$ which minimizes the left side of (4.16). The remainder of the proof is divided into two sections. First we find a minimizing $v \varepsilon R^N$ and bound the block diagonalized form of the left side of (4.16). Then we show that this bound is the diagonalized form of the right-hand side of (4.15).

Let

$$u \equiv \sum_{i=1}^{N} u_i \xi_i, \text{ and } v \equiv \sum_{i=1}^{N} v_i \xi_i.$$

Let $i \varepsilon \{ 1, 2, \ldots , (N-1)/2 \}$ and then set $j = N + 1 - i$, $x = x_i$, and $y = y_i$. Then

$$[\, QE_2Q^{-1}v \,]_i = [\, QE_2Qv \,]_i = x(v_i - v_j)$$

and

$$[\, QE_2Q^{-1}v \,]_j = [\, QE_2Qv \,]_j = y(v_j - v_i).$$

The proof is now reduced to showing the result is true for an arbitrary 2x2 block and then summing over each 2x2 block. Define

$$g(v_i,v_j) \equiv (u_i - xv_i + xv_j)^2 + (u_j - yv_j + yv_i)^2,$$

which is $(u - E_2v)^T(u - E_2v)$ restricted to a 2x2 block. Then

$$\frac{\partial}{\partial v_i} g = 2(-x)(u_i - xv_i + xv_j) + 2y(u_j - yv_i + yv_j)$$

and

$$\frac{\partial}{\partial v_j} g = - \frac{\partial}{\partial v_i} g$$

Let $\beta \equiv x^2 + y^2$. We minimize g at the (nonunique) point

$$\bar{v}_i = \beta^{-1}u_ix \text{ and } \bar{v}_j = \beta^{-1}u_jy.$$

Hence, the minimum value of g is

$$\begin{aligned} g(\bar{v}_i,\bar{v}_j) &= \beta^{-2} \{ (y^2u_i + xyu_j)^2 + (x^2u_j + xyu_i)^2 \} \\ &\le 2\beta^{-2} \{ (y^4 + x^2y^2)u_i^2 + (x^4 + x^2y^2)u_j^2 \} \\ &= 2\beta^{-1} \{ y^2u_i^2 + x^2u_j^2 \} \\ &\le 4 \{ y^2u_i^2 + x^2u_j^2 \}, \end{aligned} \tag{4.17}$$

where (4.11) and (4.14) are used to bound β^{-1}.

Q can be used to diagonalize A_k. When the right-hand side of (4.15) is restricted to the 2x2 block, it becomes

$$\frac{1}{4} h_k^4 [u_i\ u_j] \begin{vmatrix} \lambda_i & 0 \\ 0 & \lambda_j \end{vmatrix}^2 \begin{vmatrix} u_i \\ u_j \end{vmatrix}$$

$$= \frac{1}{4} [u_i\ u_j] \begin{vmatrix} 4y & 0 \\ 0 & 4x \end{vmatrix}^2 \begin{vmatrix} u_i \\ u_i \end{vmatrix}$$

$$= 4 \{ y^2 u_i^2 + x^2 u_j^2 \},$$

which is what we need to bound (4.17). Summing over each of the 2x2 blocks completes the proof.

QED

For the discrete norms we need the following norm estimate:

<u>Corollary 4.6</u>: Let $k > 1$ be an integer. For $0 \leq s \leq 2$ and $u \in M_k$,

$$\min_{v \in M_{k-1}} (u - E_2 v)^T A_k^s (u - E_2 v) \leq 4^{s-1} h_k^{4-2s} u^T A_k^2 u. \tag{4.18}$$

<u>Proof</u>: We use Lemmas 4.3 and 4.5:

$$\min_{v \in M_{k-1}} (u - E_2 v)^T A_k^s (u - E_2 v) \leq \Lambda_k^s \min_{v \in M_{k-1}} (u - E_2 v)^T (u - E_2 v)$$

$$\leq \Lambda_k^s \{ \tfrac{1}{4} h_k^4 u^T A_k^2 u \}$$

$$\leq 4^s h_k^{-2s} \{ \tfrac{1}{4} h_k^4 u^T A_k^2 u \}$$

$$= 4^{s-1} h_k^{4-2s} u^T A_k^2 u.$$

QED

We prove that (the Approximating Error Estimate) Hypothesis 2.7

holds using a duality argument when $\alpha = 1$:

<u>Lemma 4.7</u>: Let $k \geq 1$ be a fixed integer. Then

$$|||E_2\bar{q}-e_n|||_0 \leq h_k^2 |||e_n|||_2.$$

<u>Proof</u>: Let $\rho, \eta \in M_k$ satisfy

$$A_k \eta = \rho.$$

It is immediate that

$$|||\eta|||_2 = |||\rho|||_0. \tag{4.19}$$

Take $\rho = E_2\bar{q}-e_n$. Then for all $v \in M_k$,

$$(E_2\bar{q}-e_n)^T v = (A_k\eta)^T v. \tag{4.20}$$

Using $v = E_2\bar{q}-e_n$ in (4.20) and applying Corollary 4.6 (with $s = 1$) and (4.19) gives us

$$|||E_2\bar{q}-e_n|||_0^2 = a_k(\eta, E_2\bar{q}-e_n)$$

$$= a_k(\eta-E_2\omega, E_2\bar{q}-e_n), \text{ for any } \omega \in M_{k-1}$$

$$\leq \inf_{\omega \in M_{k-1}} |||\eta-E_2\omega|||\ |||E_2\bar{q}-e_n|||$$

$$\leq h_k |||\eta|||_2 |||E_2\bar{q}-e_n|||$$

$$\leq h_k |||\rho|||_0 |||E_2\bar{q}-e_n|||$$

$$= h_k |||E_2\bar{q}-e_n|||_0 |||E_2\bar{q}-e_n|||. \tag{4.21}$$

The error in the elliptic projection $|||E_2\bar{q}-e_n|||$ can be estimated.

$$|||E_2\bar{q}-e_n|||^2 = a_k(E_2\bar{q}-e_n, E_k\bar{q}-e_n)$$

$$= -a_k(E_2\bar{q}-e_n, e_n)$$

$$\leq |||E_2\bar{q}-e_n|||_0 |||e_n|||_2. \qquad (4.22)$$

Substituting (4.21) into (4.22) gives us

$$|||E_2\bar{q}-e_n||| \leq h_k |||e_n|||_2.$$

Substituting this back into (4.21) completes the proof.

QED

We can replace the smoothing step (2.7),

$$(z_i-z_{i-1},v) = \Lambda_k^{-1}[f_k(v) - a_k(z_{i-1},v)], \text{ for all } v \in M_k,$$

with

$$z_i = z_{i-1} + \Lambda_k^{-1}\{ f_k - A_k z_{i-1} \}. \qquad (4.23)$$

Also, we can replace the correction recursion step (2.8),

$$a_{k-1}(\bar{q},v) = f_k(Ev) - a(z_n,Ev), \text{ for all } v \in M_{k-1},$$

with

$$A_{k-1}\bar{q} = E^T\{ f_k - A_k z_n \} = \bar{f}_{k-1}. \qquad (4.24)$$

We now state the convergence theorem. Its proof is identical to the one for Theorem 2.9.

Theorem 4.8: Define the k-level scheme (Algorithm 2.4) using (4.23) instead of (2.7) and (4.24) instead of (2.8). Define $\alpha = 1$. Let $p > 1$ be any fixed integer. For any constant $0 < \gamma < 1$ there exists a nonnegative integer I which depends only on p and γ, such that

$$|||e_{m+n+1}||| \leq \gamma |||e_0|||, \text{ for all } m+n \geq I.$$

Further, the constant C_4 in (2.18) is equal to 4.

Before we can show that the j-level scheme (Algorithm 2.11) holds, we need to define projection and injection operators between the spaces H and M_j, $1 \leq j \leq k$:

$$p_j: H \xrightarrow[\text{onto}]{} M_j \qquad \text{and}$$

$$i_j: M_j \xrightarrow[\text{1-1}]{} H$$

(see (2.3)). We define p_j as evaluation of $u \in H$ at the grid points of Ω^j. We define i_j as piecewise-linear interpolation. These operators have properties which are worth pointing out.

Lemma 4.9: Let j be an integer. Then

(a) $p_j i_j$ = Identity on Ω^j, and

(b) $i_j E_2 p_{j-1} = i_{j-1} p_{j-1}$ on Ω^{j-1}.

To show that the j-level scheme (Algorithm 2.11) holds, the assumptions of Theorem 2.12 must be verified. Recall that for $j \geq 1$, $\tilde{u}_j$ is computed (by Algorithm 2.11) and approximates the solution u_j of the j-level problem.

Theorem 4.10: Let $j \geq 1$ be a fixed integer. Then the following holds:

(a) $|||p_j u - u_j||| \leq Kh_j, \; j \geq 1$

(b) $|||p_j u - E_2 p_{j-1} u||| \leq Kh_j, \; j \geq 2$

(c) $|||u_1 - \tilde{u}_1||| \leq Kh_1$

(d) Theorem 2.12 applies to the finite-difference case of this chapter and Algorithm 2.11 is an $O(N_j \log N_j)$ method for approximating the solution of (4.1).

Proof: (a) The proof of this requires that the solution $u \in C^4[0,1]$ and right-hand side $f \in C^2[0,1]$:

$$|||p_j u - u_j||| \leq \bar{K}h_j \|u^{(4)}\|_0 = \bar{K}h_j \|f_{xx}\|_0 = Kh_j$$

(see Varga [50]).

(b) The proof of this lies in the fact that the operator p_j is bounded in the sense that

$$a_j(p_j v, p_j v) \leq \bar{K}a(v,v), \text{ for all } v \in H,.$$

Using this reduces the proof to an interpolation estimate which requires the use of Lemma 4.9:

$$\begin{aligned} |||p_j u - E_2 p_{j-1} u||| &= |||p_j u - p_j i_j E_2 p_{j-1} u||| \\ &= |||p_j (u - i_j E_2 p_{j-1} u)||| \\ &\leq \bar{K}|||u - i_j E_2 p_{j-1} u||| \\ &= \bar{K}|||u - i_{j-1} p_{j-1} u||| \\ &\leq Kh_{j-1} \end{aligned}$$

(see Schultz [43]).

(c) Since the one-level problem is solved directly, we assume that it is approximated to the order of truncation.

(d) Parts (a)-(c) are the requirements of Theorem 2.12. The operation count comes from (2.20) with $\sigma = 2$.

QED

We conclude this section by considering (4.1) without the restriction of $P = 1$ and $S = 0$. We discretize (4.1) by central differences to get an $N_k \times N_k$ linear system of equations

$$\bar{A}_k u_k = f_k.$$

As before, we define

$$\bar{A}_{k-j} = E_2^T \bar{A}_{k-j+1} E_2, \; 1 \leq j < k, \text{ and}$$

$$\tilde{a}_j(u,v) = u^T \bar{A}_j v, \; 1 \leq j \leq k.$$

Once again, half the rows and columns of $\bar{A}_{k-1}$ are zero. We can re-order the matrix so that the first N_{k-1} the rows and columns are tridiagonal and the remaining rows and columns are zero. We define $|||u|||^2 = \tilde{a}_j(u,v)$ and the discrete norms $|||u|||_s$, $0 \leq s \leq 2$ and $u \in M_j$, are defined as usual. We can prove the following:

<u>Theorem 4.11</u>: Let $j \geq 1$ be a fixed integer. Then the following holds:

(a) $\tilde{A}_j$ is symmetric and has N_j positive eigenvalues which are bounded by Ch_j^{-2}, where C is independent of j.

(b) For $\beta = 0$, 1, or 2, there exist positive constants $C_{1,\beta}$ and $C_{2,\beta}$ such that

$$C_{1,\beta} \, |||u|||_\beta \leq |||u|||_\beta \leq C_{2,\beta} \, |||u|||_\beta \text{ for all } u \in M_j.$$

(c) $|||E_2\bar{q}-e_n|||_0 \leq C_{1,2}^{-1} \, h_k^2 \, |||e_n|||_2$.

<u>Proof</u>: Let $j \geq 1$ be a fixed integer.

(a) This is proved by direct computation.

(b) The proof for each of the three cases $\beta = 0$, 1, or 2 is different and increasingly more complex. Recall from (4.2) that constants $\underline{p}$, $\bar{p}$, $\underline{s}$, and $\bar{s}$ exist such that

$$\underline{p} \leq P(x) \leq \bar{p}, \ \underline{s} \leq S(x) \leq \bar{s}, \text{ for all } x \in \bar{\Omega}.$$

The proof of the case $\beta = 2$ requires that a constant $\bar{p}'$ exist such that

$$| \, P_x(x) \, | \leq \bar{p}'.$$

The first two cases are straightforward. When $\beta = 0$, it is immediate that

$$C_{1,1} = C_{2,1} = 1.$$

When $\beta = 1$, Stiefel [47] proves that

$$C_{1,1} = (\underline{p} + c\underline{s})^{-1} \quad \text{and} \quad C_{2,1} = \bar{p} + c\bar{s},$$

where c is a constant independent of h_j.

The proof for the case of $\beta = 2$ follows the argument for the continuous case (see Schultz [43]). Let

$$D_+u_j = u_{j+1} - u_j \quad \text{and}$$

$$D_-u_j = u_j - u_{j-1}$$

be forward and backward difference operators. By direct computation we can show that

$$\frac{1}{2} \, |||D_-u|||_0 \leq |||u|||_0 \leq \frac{1}{2} \, |||D_-u|||_0. \tag{4.25}$$

The constants in (4.25) are not optimal, but they are independent of h_j. After discretizing (4.1), we are left with a system of equations of the form

$$-D_+P_{i-1/2}D_-u_i + S_iu_i = f_i, \tag{4.26}$$

where u is a grid function on Ω^j which vanishes at the boundary points. We will drop the subscripts from the remainder of the proof. Note that

$$D_+PD_-u = PD_+D_-u + (D_+P)(D_+u). \tag{4.27}$$

Substituting (4.27) into (4.26) and re-arranging terms yields

$$-D_+D_-u = \frac{1}{P}\{ (D_+P)(D_+u) - Su + f \}.$$

So,

$$|||D_+D_-u|||_0 \le \underline{p}^{-1}\{ \bar{p}'|||D_+u|||_0 + \bar{s}|||u|||_0 + |||f|||_0 \}. \tag{4.28}$$

Since our problem is elliptic, we know that

$$\begin{aligned} |||D_+u|||_0^2 &\le \underline{p}^{-1}\, ||||u||||^2 \\ &= \underline{p}^{-1}\,(f,u)_j \\ &\le \underline{p}^{-1}|||f|||_0|||u|||_0 \\ &\le (2\underline{p})^{-1}|||f|||_0|||D_+u|||_0. \end{aligned}$$

Hence,

$$|||D_+u|||_0 \le (2\underline{p})^{-1}\,|||f|||_0. \tag{4.29}$$

Using (4.25) and (4.29) gives us

$$|||u|||_0 \le \tfrac{1}{2}|||D_+u|||_0 \le (2\underline{p})^{-1}\,|||f|||_0. \tag{4.30}$$

Substituting (4.29) and (4.30) into (4.28) yields

$$\begin{aligned} |||D_+D_-u|||_0 &\le \underline{p}^{-1}\{ (2\underline{p})^{-1}(\bar{p}' + \bar{s}) + 1 \}\,|||f|||_0 \\ &= C_{1,2}^{-1}\,|||f|||_0. \end{aligned}$$

Since $|||u|||_2 = |||D_+D_-u|||_0$ and $||||u||||_2 = |||f|||_0$, we have

$$|||u|||_2 \le C_{1,2}^{-1}\,||||u||||_2,$$

where $C_{1,2}$ is independent of h_j. An easier argument proves that

$$C_{2,2} = \bar{p} + \tfrac{1}{2}(\bar{p}' + \bar{s}).$$

This completes the proof of (b).

(c) Apply part (b) to Lemma 4.7.

QED

Theorems 4.8 and 4.10 can be rewritten using the $||||\cdot||||$ norm instead of the $|||\cdot|||$ norm. This proves that the variable coefficient Dirichlet problem (4.1) can be solved using the theory of Chapter 2 using $O(N_k \log N_k)$ operations.

4.3 Two Levels

In this section, we give a summary of sharper results that can be proved when Algorithm 2.4 is restricted to two grids. The proofs can be found in Douglas [19]. We first investigate how the order of interpolation affects the rate of convergence. This can be used to optimize the number of smoothing iterations and correction recursions to minimize the amount of work required. We restrict our attention to $P = 1$ and $S = 0$ in (4.1) and the cases of piecewise-linear and piecewise-cubic interpolation between grids.

As in Lemma 2.8, the initial error e_0 can be expanded in terms of the eigenvectors:

$$e_0 = \sum_{i=1}^{N} \beta_i\, \xi_i.$$

The eigenvector expansion of e_{m+n+1} can be written in terms of an iteration matrix $M_{m,n,i,j}$ and the initial error e_0:

$$e_{m+n+1} = M_{m,n,i,j} e_0. \tag{4.31}$$

The iteration matrix $M_{m,n,i,j}$ has four parameters:

m = number of smoothing iterations after solving the correction recursion problem

n = number of smoothing iterations before solving the correction recursion problem

i = either 2 or 4: whether E_2^T or E_4^T is used to project onto the coarse grid

j = either 2 or 4: whether E_2 or E_4 is used to interpolate onto the fine grid

The four iteration matrices of interest are denoted by

linear-linear:	$M_{m,n,2,2}$
linear-cubic:	$M_{m,n,2,4}$
cubic-linear:	$M_{m,n,4,2}$
cubic-cubic:	$M_{m,n,4,4}$

The convergence rate of the two-level version of Algorithm 2.4 is determined by analyzing the spectral radius of each iteration matrix in (4.31). Let

$\rho(M)$ ≡ spectral radius of a matrix M.

We can show that

$$\rho(M_{0,m+n,i,j}) = \rho(M_{m,n,i,j}).$$

and

$$\rho(M_{0,n,i,j}) \sim \begin{cases} 2^{-n}, & \text{if } 1 \le n \le n_0 \\ \dfrac{Kc^p}{(n+p)^p}, & \text{if } n > n_0,\ c \to e^{-1} \text{ as } n \to \infty, \end{cases}$$

where

	n_0	K	p
linear-linear	3	1	1
linear-cubic	4	12	2
cubic-linear	4	12	2
cubic-cubic	4	1	1

(4.32)

When the number of smoothing iterations is three or less, linear-linear is the most efficient to use. When the number of smoothing iterations is quite large, either linear-cubic or cubic-linear is the most efficient.

Using the spectral radii of the iteration matrices, we can choose the number of smoothing iterations and correction recursions so as to minimize the amount of work required to reduce the error by a factor of $0 < \varepsilon < 1$. If each two-level iteration reduces the error by a factor of $\rho(n)$, where n is the number of smoothing iterations, then we want to do r correction recursions so that

$$\rho^r(n) \le \varepsilon$$

or

$$r \geq \frac{\ln \varepsilon}{\ln \rho(n)}. \tag{4.33}$$

We can prove that the optimal choice of n in (4.33) is

$$n = n_0,$$

where n_0 is defined in (4.32) for each iteration matrix.

CHAPTER 5

Two-Dimensional Finite-Difference Analysis

5.1 Introduction

In this chapter, we discuss the iterative solution of large sparse linear systems which arise in connection with finite-difference procedures for approximating the solution of the model problem:

$$\begin{cases} -(Pu_x)_x - (Pu_y)_y + Su = f \text{ in } \Omega \equiv (0,1)\text{x}(0,1) \\ \qquad\qquad\qquad\qquad\quad u = 0 \text{ on } \partial\Omega. \end{cases} \tag{5.1}$$

We assume that $f \in L^2(\Omega)$ and there exist positive constants $\underline{p}$, $\bar{p}$, $\underline{s}$, and $\bar{s}$ such that

$$\underline{p} \leq P(x) \leq \bar{p}, \ \underline{s} \leq S(x) \leq \bar{s}, \text{ for all } x \in \bar{\Omega}.$$

We seek a weak form solution of (5.1): find $u \in H_0^1$ such that

$$a(u,v) = (f,v) \quad \text{for all } v \in H,$$

where

$$a(u,v) \equiv \int_\Omega P(u_x v_x + u_y v_y) + Suv \, dx \qquad \text{and}$$

$$(f,v) \equiv \int_\Omega fv \, dx.$$

Then there exists a unique weak solution $u \varepsilon H(\Omega)$ for all $f \varepsilon L^2(\Omega)$ (see Ciarlet [16]). The theory in this chapter restricts our attention to problems for which the solution lies in

$$H = H_0^1 \cap H^2.$$

The bilinear form $a(\cdot,\cdot)$ induces the energy norm $|||u|||^2 = a(u,u)$.

Finite-difference approximations to (5.1) are solved by a variety of methods. Depending on the values of P and S, these include the fast Fourier transform, cyclic reduction, sparse Gaussian elimination, preconditioned conjugate gradient, and SSOR. None of these methods is optimal order. In Section 4.2, we show that the multi-grid algorithms of Chapter 2 are optimal order for this problem. We analyze this problem in two parts. First, the case of $P = 1$ and $S = 0$ in (5.1) is considered. Then we analyze the general case. In Section 5.3, we consider the case when only two grids are used. We present a summary of sharper results that are proved in Douglas [19]. We first investigate how the order of interpolation affects the rate of convergence. We restrict our attention to the cases of piecewise-linear and piecewise-cubic interpolation between grids. This can be used to optimize the number of smoothing iterations and correction recursions to minimize the amount of work required.

5.2 K Levels

In this section, we assume that the number of grids is arbitrarily large. We prove that the theorems and algorithms of Chapter 2 apply to a particular discretization of (5.1). Most of this section contains

analysis of the Dirichlet Poisson problem. At the end of this section, we extend the analysis to the separable elliptic problem (5.1).

Let $k > 1$ and $\bar{N}_0 > 0$ be fixed integers. We define $\bar{N}_j = 2\bar{N}_{j-1} + 1$, $j \geq 1$. We define k uniform grids Ω_2^j, $1 \leq j \leq k$, as products of the one-dimensional domains Ω^j:

$$h_k = (\bar{N}_k + 1)^{-1},$$

$$h_j = 2^{k-j}h_k, \quad \text{and}$$

$$\Omega_2^j = \Omega^j \times \Omega^j.$$

Each grid Ω_2^j covers the interior of Ω with $N_j = \bar{N}_j^2$ points. Following (2.2), we know that $\sigma = 4$.

For the moment, we assume that $P = 1$ and $S = 0$ in (5.1). The general case will be analyzed at the end of the chapter. We discretize (5.1) by central differences to obtain a system of linear equations

$$B_k u_k = h_k^2 f_k, \tag{5.2}$$

where u_k and f_k approximate the solution u and the right-hand side f on Ω_2^k. The $N_j \times N_j$ matrix B_k is given by

$$B_k = h_k \{A_k \otimes I_N + I_N \otimes A_k\},$$

where I_N is the $N_k x N_k$ identity matrix and A_k is defined in (4.4).

We define solution spaces M_j, $1 \leq j \leq k$, as the tensor-product of the one-dimensional solution spaces defined in (4.5). Following the techniques of Chapter 4, we define interpolation operators between adjacent solution spaces

$$E: M_{j-1} \to M_j \quad \text{and}$$

$$R = E^T: M_j \to M_{j-1}.$$

The piecewise-bilinear and piecewise-bicubic interpolation matrices we are interested in are tensor-products of their one-dimensional equivalents:

$$E_b = E_2 \otimes E_2 \quad \text{and}$$

$$E_c = E_4 \otimes E_4.$$

For the multi-grid analysis of Chapter 2, piecewise-bilinear interpolation will suffice. In Section 5.3, we will analyze the effect of bilinear and cubic interpolation on the order of convergence of the multi-grid algorithms of Chapter 2 when they are restricted to two levels.

Problems on coarser grids are defined using the interpolation matrices. Linear systems on coarser grids, B_j, $1 \leq j < k$, are defined recursively:

$$B_{k-j} = E_b^T B_{k-j+1} E_b.$$

Let $N = N_k$ and $r \in R^N X R^N$ be a right-hand side for level k. Then the (k-1)-level problem is defined by

$$B_{k-1} \bar{q} = E^T r = f_{k-1}, \quad \bar{q} \in M_{k-1}, \tag{5.3}$$

where E is either the linear or cubic interpolant. For the analysis of

Chapter 2, E in (5.3) is the bilinear interpolant. Three-fourths of the rows and columns of B_{k-1} are zero. We can re-order the matrix so that the nonzero rows and columns are the first N_{k-1} rows and columns. Then B_{k-1} has a submatrix whose form is identical to B_k. The (k-i)-level problems are defined similarly to (5.3).

Before the theory of Chapter 2 can be applied, we must complete the definitions required for the triples used by Problems 2.1 and 2.2. For $j \geq 1$, a bilinear form $a_j(\cdot,\cdot)$, a linear functional $f_j(v)$, and an inner product $(\cdot,\cdot)_j$ are defined similarly to (4.8):

$$a_j(u,v) = u^T B_j v$$

$$f_j(v) = h_j^2 f_j^T v, \text{ for all } u,v \in M_j, \quad \text{and}$$

$$(u,v)_j = h_j^2 u^T v.$$

The bilinear form $a_j(\cdot,\cdot)$ induces the energy norm

$$|||u|||^2 = a_j(u,u)$$

for all $u \in M_j$. Associated with each $a_j(\cdot,\cdot)$ are N_j nonzero eigenvalues $\lambda_{\rho\omega}$ and eigenvectors $\xi_{\rho\omega}$ satisfying (2.6). Let Λ_j be the largest eigenvalue. For $-2 \leq s \leq s$, discrete norms $|||v|||_s$ are defined similarly to (4.9).

In order to establish the convergence of Algorithms 2.4 and 2.11 for the finite-difference case of this chapter, we must verify Hypotheses 2.3, 2.6, and 2.7. We begin by showing that (the Energy Norm Consistency) Hypothesis 2.3 holds. It is derived using the definition of the linear systems B_{j-1}, $j > 1$. The proof is virtually identical to the one for Lemma 4.2.

<u>Lemma 5.1</u>: Let $j > 1$ be an integer. Then

$$a_j(E_b v, E_b w) \quad = \quad a_{j-1}(v,w), \text{ for all } v,w \in M_{j-1}.$$

That (the Maximum Eigenvalue) Hypothesis 2.6 holds is a simple consequence of the explicit formula of the eigenvalues of B_j.

<u>Lemma 5.2</u>: $\Lambda_j \;\le\; 8h_j^{-2}$.

<u>Proof</u>: The eigenvalues of $a_j(\cdot,\cdot)$ are given by

$$\lambda_{\rho\omega} \quad = \quad \lambda_\rho + \lambda_\omega,\ 1 \le \rho,\omega \le N_j,$$

where λ_ρ and λ_ω are eigenvalues of the one-dimensional matrix A_j (see Isaacson and Keller [32]). The result follows from Lemma 4.3.

QED

In order to prove that (the Approximating Error Estimate) Hypothesis 2.7 holds, we must prove two auxiliary norm estimates first. These estimates rely on the fact that the linear and cubic interpolants E_b and E_c can be block diagonalized using a sine transform. Let

$$Q_2 \quad = \quad Q \otimes Q,$$

where Q is defined in (4.10). Note that Q_2 is an orthogonal matrix.

<u>Lemma 5.3</u>: Set E to either E_b or E_c and $\bar{E}$ to the one-dimensional equivalent of E. Then

$$Q_2 E Q_2^{-1} \quad = \quad Q_2 E Q_2 \quad = \quad (Q\bar{E}Q) \otimes (Q\bar{E}Q).$$

<u>Proof</u>: Apply the matrix tensor-product identity

$$(M_1 \otimes M_2)\,(M_3 \otimes M_4) \quad = \quad (M_1\, M_3) \otimes (M_2\, M_4)$$

twice, where the M_i are appropriate matrices.

QED

We can visualize $Q_2 E_b Q_2^{-1}$ by

$$Q_2 E_b Q_2^{-1} \quad = \quad \left|\begin{array}{cccccc} x_1 QE_2Q & & & & & -x_1 QE_2Q \\ & x_2 QE_2Q & & & -x_2 QE_2Q & \\ & & \ddots & \iddots & & \\ & & \iddots & \ddots & & \\ & -y_2 QE_2Q & & & y_2 QE_2Q & \\ -y_1 QE_2Q & & & & & y_1 QE_2Q \end{array}\right|,$$

where x_i and y_i, $i = 1, 2, \ldots, N_{k-1}$, are defined by (4.11).

We now prove the two auxiliary norm estimates.

<u>Lemma 5.4</u>: Let $k \ge 1$ be a fixed integer, $N = N_k$, and $u \in M_k$. Then

$$\min_{v \in M_{k-1}} (u - E_b v)^T (u - E_b v) \;\le\; h_k^4\, u^T B^2\, u. \qquad (5.4)$$

<u>Proof</u>: Let $N = N_k$. As in Lemma 4.5

$$\min_{v \varepsilon M_{k-1}} (u - E_2 v)^T (u - E_2 v) \quad = \quad \min_{v \varepsilon R^N \chi R^N} (u - E_2 v)^T (u - E_2 v). \qquad (5.5)$$

This is because the elements of $v \varepsilon M_{k-1}$ which are required to be zero correspond to the columns of E_b which are zero. It turns out that finding a $v \varepsilon R^N \chi R^N$ which minimizes the right-hand side of (5.5) is easier than finding a $v \varepsilon M_{k-1}$ which minimizes the left side of (5.5). The remainder of the proof is divided into two sections. First we find a minimizing $v \varepsilon R^N \chi R^N$ and bound the block diagonalized form of the left side of (5.5). Then we show that this bound is the diagonalized form of the right-hand side of (5.4).

Let

$$u = \sum_{i,j=1}^{N} u_{ij}\, \xi_{ij} \quad \text{and} \quad v = \sum_{i,j=1}^{N} v_{ij}\, \xi_{ij}$$

Let $i, j \varepsilon \{1, 2, \ldots, (N-1)/2\}$ and set $\eta = N + 1 - i$ and $\rho = N + 1 - j$. Define

$$\begin{aligned} a &= x_i\, x_j, & c &= y_i\, x_j, \\ b &= x_i\, y_j, \quad \text{and} & d &= y_i\, y_j. \end{aligned} \qquad (5.6)$$

Then the nonzero (i,j)th 4x4 block of $Q_2 E_b Q_2$ is

$$\begin{vmatrix} a & -a & -a & a \\ -b & b & b & -b \\ -c & c & c & -c \\ d & -d & -d & d \end{vmatrix}.$$

Let u_i, u_η, u_j, and u_ρ be the elements of u corresponding to the 4x4 block of $Q_2 E_b Q_2$ (and similarly for v). Define

$$\begin{aligned} g(u,v) \quad = \quad & (u_i - a(v_i - v_\eta - v_j + v_\rho))^2 + \\ & (u_\eta + b(v_i - v_\eta - v_j + v_\rho))^2 + \\ & (u_\eta + c(v_i - v_\eta - v_j + v_\rho))^2 + \\ & (u_\eta - d(v_i - v_\eta - v_j + v_\rho))^2 \end{aligned}$$

which is $(u - E_b v)^T (u - E_b v)$ restricted to a 4x4 block. Then

$$\begin{aligned} \frac{\partial}{\partial v_i} g \;=\; \frac{\partial}{\partial v_\rho} g \;=\; & -2a(u_i - a(v_i - v_\eta - v_j + v_\rho)) \\ & + 2b(u_\eta + b(v_i - v_\eta - v_j + v_\rho)) \\ & + 2c(u_\eta + c(v_i - v_\eta - v_j + v_\rho)) \\ & - 2d(u_\eta - d(v_i - v_\eta - v_j + v_\rho)) \end{aligned}$$

and

$$\frac{\partial}{\partial v_\eta} g \;=\; \frac{\partial}{\partial v_j} g \;=\; -\frac{\partial}{\partial v_i} g$$

Let $\beta = a^2 + b^2 + c^2 + d^2$. Then g is minimized at the (nonunique) point

$$\begin{aligned} \bar{v}_i &= \beta^{-1} a u_i, & \bar{v}_j &= \beta^{-1} c u_j, \\ \bar{v}_\eta &= \beta^{-1} b u_\eta, \quad \text{and} & \bar{v}_\rho &= \beta^{-1} d u_\rho. \end{aligned}$$

Hence, the minimum value of g is

$$\begin{aligned} g(u,\bar{v}) \; = \; & (u_i - a\beta^{-1}(au_i - bu_\eta - cu_j + du_\rho))^2 + \\ & (u_\eta + b\beta^{-1}(au_i - bu_\eta - cu_j + du_\rho))^2 + \\ & (u_\eta + c\beta^{-1}(au_i - bu_\eta - cu_j + du_\rho))^2 + \\ & (u_\eta - d\beta^{-1}(au_i - bu_\eta - cu_j + du_\rho))^2 \end{aligned}$$

$$\begin{aligned} \le \; 4\beta^{-2} \{ \; & (b^2 + c^2 + d^2)^2u_i^2 + a^2b^2u_\eta^2 + a^2c^2u_j^2 + a^2d^2u_\rho^2 + \\ & a^2b^2u_i^2 + (a^2 + c^2 + d^2)^2u_\eta^2 + b^2c^2u_j^2 + b^2d^2u_\rho^2 + \\ & a^2c^2u_i^2 + b^2c^2u_\eta^2 + (a^2 + b^2 + d^2)^2u_j^2 + c^2d^2u_\rho^2 + \\ & a^2d^2u_i^2 + b^2d^2u_\eta^2 + c^2d^2u_j^2 + (a^2 + b^2 + c^2)^2u_\rho^2 \; \} \end{aligned}$$

$$\begin{aligned} \le \; 4\beta^{-1} \{ \; & [b^2 + c^2 + d^2]u_i^2 + [a^2 + c^2 + d^2]u_\eta^2 + \\ & [a^2 + b^2 + d^2]u_j^2 + [a^2 + b^2 + c^2]u_\rho^2 \; \} \end{aligned}$$

$$\begin{aligned} \le \; 4\beta^{-1} \{ \; & [d^2 + y_i^2 + y_j^2]u_i^2 + [c^2 + y_i^2 + x_j^2]u_\eta^2 + \\ & [b^2 + x_i^2 + y_j^2]u_j^2 + [a^2 + x_i^2 + x_j^2]u_\rho^2 \; \} \end{aligned}$$

Since $\beta^{-1} \le 4$ and $a^2 \le a$ (and similarly for b, c, and d), we get

$$\begin{aligned} g(u,\bar{v}) \; \le \; 16 \{ \; & [2d + y_i^2 + y_j^2]u_i^2 + [2c + y_i^2 + x_j^2]u_\eta^2 + \\ & [2b + x_i^2 + y_j^2]u_j^2 + [2a + x_i^2 + x_j^2]u_\rho^2 \; \}. \end{aligned} \qquad (5.7)$$

Q_2 can be used to diagonalize B_k. When the right-hand side of (5.4) is restricted to the 4x4 block, it becomes

$$[u_i \; u_\eta \; u_j \; u_\rho] \left| \begin{matrix} 4(y_i+y_j) & & & \\ & 4(y_i+x_j) & & 0 \\ & 0 & 4(x_i+y_j) & \\ & & & 4(x_i+x_j) \end{matrix} \right|^2 \left| \begin{matrix} u_i \\ u_\eta \\ u_j \\ u_\rho \end{matrix} \right|$$

$$\begin{aligned} = \; 16 \{ \; & (y_i + y_j)^2u_i^2 + (y_i + x_j)^2u_\eta^2 + (x_i + y_j)^2u_j^2 + \\ & (x_i + x_j)^2u_\rho^2 \; \} \end{aligned}$$

$$\begin{aligned} = \; 16 \{ \; & [2d + y_i^2 + y_j^2]u_i^2 + [2c + y_i^2 + x_j^2]u_\eta^2 + \\ & [2b + x_i^2 + y_j^2]u_j^2 + [2a + x_i^2 + x_j^2]u_\rho^2 \; \}, \end{aligned}$$

which is what we needed to bound (5.7). Summing over each 4x4 block gives us the result.

QED

For the discrete norms we need the following bound, whose proof is virtually identical to the one for Corollary 4.6.

Corollary 5.5: Let $k \ge 1$ be a fixed integer and $N = N_k$. For $0 \le s \le 2$ and $u \in M_k$,

$$\min_{v \in M_{k-1}} (u - E_b v)^T B^s (u - E_b v) \; \le \; 8^s h_k^{4-2s} u^T B^2 u. \qquad (5.8)$$

A duality argument proves that (the Approximating Error Estimate) Hypothesis 2.7 holds when $\alpha = 1$. The proof is virtually identical to the one for Lemma 4.7.

<u>Lemma 5.6</u>: Let $k \geq 1$ be a fixed integer. Then

$$|||E_b\bar{q}-e_n|||_0 \leq 8^{1/2}\ h_k^2\ |||e_n|||_2.$$

We can replace the smoothing step (2.7),

$$(z_i-z_{i-1},v) = \Lambda_k^{-1}[f_k(v) - a_k(z_{i-1},v)],\ \text{for all } v \in M_k,$$

with

$$z_i = z_{i-1} + \Lambda_k^{-1}\{\ f_k - B_k z_{i-1}\ \}. \tag{5.9}$$

Also, we can replace the correction recursion step (2.8),

$$a_{k-1}(\bar{q},Ev) = f_k(v) - a(z_n,Ev).$$

with

$$B_{k-1}\bar{q} = E^T\{\ f_k - B_k z_n\ \} = \bar{f}_{k-1}. \tag{5.10}$$

We now state the convergence theorems for the case of model problem (5.1). Their proofs are virtually identical to the ones for Theorems 2.9 and 4.10.

<u>Theorem 5.7</u>: Define the k-level scheme (Algorithm 2.4) using (5.9) instead of (2.7) and (5.10) instead of (2.8). Define $\alpha = 1$. Let $p > 1$ be any fixed integer. For any constant $0 < \gamma < 1$ there exist a nonnegatives integer I which depends only on p and γ, such that

$$|||e_{m+n+1}||| \leq \gamma|||e_0|||,\ \text{for all } m+n \geq I.$$

Further, the constant C_4 in (2.18) is equal to $16\cdot 2^{1/2}$.

As will be seen in Section 5.3, the constant C_4 is considerably overestimated in Theorem 5.7.

<u>Theorem 5.8</u>: Theorem 2.12 applies to the finite-difference case of this chapter and Algorithm 2.11 is an $O(N_j)$ method for approximating the solution of (4.1).

We conclude this section by considering (5.1) when it is no longer restricted to $P = 1$ and $S = 0$. We discretize (5.1) by central differences to get an $N_k \times N_k$ linear system of equations

$$\tilde{B}_k u_k = f_k.$$

As before, we define

$$\tilde{B}_{k-j} = E_b^T\ \tilde{B}_{k-j+1}\ E_b,\ 1 \leq j < k,\ \text{ and}$$

$$\tilde{a}_j(u,v) = u^T\ \tilde{B}_j\ v,\ 1 \leq j \leq k.$$

Once again, three-fourths of the rows and columns of $\tilde{B}_{k-1}$ are zero. We can re-order the matrix so that the first N_{k-1} rows and columns are nonzero and the remaining rows and columns are zero. We define

$|||u|||^2 = \tilde{a}_j(u,v)$ and the discrete norms $|||u|||_s$, $0 \le s \le 2$ and $u \in M_j$, are defined as usual. The following is the two-dimensional analogue of Theorem 4.11.

Theorem 5.9: Let $j \ge 1$ be a fixed integer. Then the following holds:
(a) $\tilde{B}_j$ is symmetric and has N_j positive eigenvalues which are bounded by Ch_j^{-2}, where C is independent of j.
(b) For $\beta = 0, 1,$ or 2, there exist positive constants C_1 and C_2 such that

$$C_{1,\beta} ||| u |||_\beta \le |||u|||_\beta \le C_{2,\beta} ||| u |||_\beta \text{ for all } u \in M_j.$$

(c) $|||E_b\bar{q}-e_n|||_0 \le 8^{1/2} C_{2,1} h_k^2 |||e_n|||_2$.

Proof: Let $j \ge 1$ be a fixed integer.
(a) This is proved by direct computation.
(b) This proof is similar to the proof of part (b) of Theorem 4.11. In particular, if there exists a constant $\bar{p}'$ such that

$$0 \le |\max\{P_x, P_y\}| \le \bar{p}',$$

then

$$C_{1,0} = C_{2,0} = 1,$$

$$C_{1,1} = (\underline{p} + c\underline{s})^{-1}, \quad C_{2,1} = \bar{p} + c\bar{s},$$

$$C_{2,1} = \underline{p}\,[(\underline{p})^{-1}(\bar{p}' + \bar{s}) + 1]^{-1}, \quad \text{and}$$

$$C_{2,2} = \bar{p} + \bar{p}' + \bar{s}.$$

where c is independent of h_j.

(c) Apply part (b) to Lemma 5.6.

QED

Theorems 5.7 and 5.8 can be rewritten using the $|||\cdot|||$ norm instead of the $|||\cdot|||$ norm. This proves that Algorithm 2.11 is optimal order for solving the variable coefficient Dirichlet problem (5.1).

5.3 Two Levels

In this section, we give a summary of sharper results that can be proved when Algorithm 2.4 is restricted to two grids. The proofs can be found in Douglas [19]. We first investigate how the order of interpolation affects the rate of convergence. This can be used to optimize the number of smoothing iterations and correction recursions to minimize the amount of work required. We restrict our attention to P = 1 and S = 0 in (5.1) and the cases of piecewise-linear and piecewise-cubic interpolation between grids.

As in Lemma 2.8, the initial error e_0 can be expanded in terms of the eigenvectors:

$$e_0 = \sum_{i=1}^{N} \beta_i \xi_i.$$

The eigenvector expansion of e_{m+n+1} can be written in terms of an iteration matrix $M_{m,n,i,j}$ and the initial error e_0:

$$e_{m+n+1} = M_{m,n,i,j}\, e_0. \tag{5.11}$$

The iteration matrix $M_{m,n,i,j}$ has four parameters:

m = number of smoothing iterations after solving the correction recursion problem

n = number of smoothing iterations before solving the correction recursion problem

i = either b or c: whether E_b^T or E_c^T is used to project onto the coarse grid

j = either b or c: whether E_b or E_c is used to interpolate onto the fine grid

The four iteration matrices of interest are denoted by

bilinear-bilinear:	$M_{m,n,b,b}$
bilinear-bicubic:	$M_{m,n,b,c}$
bicubic-bilinear:	$M_{m,n,c,b}$
bicubic-bicubic:	$M_{m,n,c,c}$

The convergence rate of the two-level version of Algorithm 2.4 is determined by analyzing the spectral radius of each iteration matrix in (5.11). Let

$\rho(M)$ = spectral radius of a matrix M.

We can show that

$$\rho(M_{0,n,i,j}) \sim \begin{cases} (.75)^n, & \text{if } 1 \le n \le n_0 \\ \dfrac{Kc}{(n+2)}, & \text{if } n > n_0,\ c \to e^{-1} \text{ as } n \to \infty, \end{cases}$$

where

	n_0	K
bilinear-bilinear	7	2
bilinear-bicubic	10	1
bicubic-bilinear	10	1
bicubic-bicubic	8	3

(5.12)

When the number of smoothing iterations is three or less, bilinear-bilinear is the most efficient to use. When the number of smoothing iterations is quite large, either linear-cubic or bicubic-bilinear is the most efficient. Bicubic-bicubic is the least efficient choice no matter how many smoothing iterations are used.

Using the spectral radii of the iteration matrices, we can choose the number of smoothing iterations and correction recursions so as to minimize the amount of work required to reduce the error by a factor of $0 < \varepsilon < 1$. If each two-level iteration reduces the error by a factor of $\rho(n)$, where n is the number of smoothing iterations, then we want to do r correction recursions so that

$$\rho^r(n) \le \varepsilon$$

or

$$r \ge \frac{\ln \varepsilon}{\ln \rho(n)}. \tag{5.13}$$

We can prove that the optimal choice of n in (5.13) is

$$n = n_0,$$

where n_0 is defined in (5.12) for each iteration matrix.

CHAPTER 6

Computational Aspects

6.1 Introduction

We have written a FORTRAN program which implements a class of finite-difference multi-grid algorithms for approximating the solution of elliptic boundary-value problems of the form

$$\begin{cases} (P(x,y)u_x)_x + (Q(x,y)u_y)_y + V(x,y)u_x + \\ \qquad\qquad W(x,y)u_y + S(x,y)u = F(x,y) \text{ in } \Omega \\ \beta(x,y)u_n + \alpha(x,y)u = \gamma(x,y) \text{ on } \partial\Omega, \end{cases} \tag{6.1}$$

where Ω is a rectangular region with boundary $\partial\Omega$. If a uniform mesh with spacing h is applied to Ω and a modified upwind discretization [6] of (6.1) is used, then we can define the approximate problem

$$\bar{\Omega}^h = \{(x+(i-1)h, y+(j-1)h),\ i = 1, \ldots, n_x^h,\ j = 1, \ldots, n_y^h\},$$

$$u_{ij}^h \cong u(x_i,y_j), \quad \text{for } (x_i,y_j) \in \Omega^h,$$

$$F_{ij}^h \cong F(x_i,y_j), \quad \text{for } (x_i,y_j) \in \Omega^h,$$

and

$$\begin{aligned} A^h_{ij}u^h_{ij} &\equiv L^h_{ij}u^h_{i-1,j} + R^h_{ij}u^h_{i+1,j} + B^h_{ij}u^h_{i,j-1} \\ &\quad + T^h_{ij}u^h_{i,j+1} + D^h_{ij}u^h_{ij} \qquad (6.2) \\ &= h^2 F^h_{ij}, \qquad i = 1, \ldots, n^h_x, \\ &\qquad\qquad\qquad j = 1, \ldots, n^h_y, \\ &= f^h_{ij}. \end{aligned}$$

When the solution is sufficiently smooth, this provides an $O(h^2)$ approximation to the solution of (6.1) (see Varga [50]).

Figure 6-1 contains a definition of the class of <u>fixed multi-grid algorithms</u> implemented in the code. This class has nine arguments:

KLevel(MaxLev, CycleC, CycleF, SmB, SmN, SmL, Smlopt, Hopt, StLev),

- MaxLev	Maximum number of levels allowed.
- CycleC	The number of correction iterations used from all but level MaxLev.
- CycleF	The number of correction iterations used from level MaxLev.
- SmB	The number of smoothing iterations performed before the first correction.
- SmL	The number of smoothing iterations performed after the last correction.
- SmN	The number of smoothing iterations performed on iterations when SmB or SmL do not take precedence.
- Smlopt	Whether smoothing iterations are to performed on the coarsest grid or not (versus a direct solve).
- Hopt	Whether we are using the scheme H (see Definition 6.3).
- StLev	Which level to begin the algorithm on. This is 1 for most of the schemes defined in this section.

Figure 6-1: Generalized Multi-Grid Algorithm

```
KLevel(MaxLev, CycleC, CycleF, SmB, SmN, SmL, Smlopt, Hopt, StLev)
|
| Generate f(StLev) and A(i), i = 1, ... , StLev
| Set CycleM(i) = CycleC, i = 1, ... , StLev
| CycleM(1) = 1 ; CycleM(MaxLev) = CycleF
| Level = LevelM = StLev
| Work(Level) = False ; Cycle(Level) = 1
|
| S: If Level = 1 and Smlopt = False          SMOOTHING
|      Then Solve A(1)u(1) = f(1) directly
|        |__ Go to I
|      Else M = SmN
|        |  If Cycle(Level) = 0 Then M = SmB
|        |  If Cycle(Level) = CycleM(Level) Then M = SmL
|        |  If M > 0
|        |    Then Work(Level) = True
|        |      |__ Do M smoothing iterations
|        |  Cycle(Level) = Cycle(Level) + 1
|        |__ If Cycle(Level) > CycleM(Level) Then Go to I
|
| C: Lev = Level - 1                          CORRECTION
|    Work(Lev) = False ; Cycle(Lev) = 0
|    Zero u(Lev)
|    If Work(Level) = True
|      Then Compute the residuals r(Level)
|        |__ Project weighted r(Level) as f(Lev)
|      Else Project weighted f(Level) as f(Lev)
|    If Level = MaxLev and Cycle(Level) = CycleM(Level)
|       and Hopt = True
|      Then CycleM(J) = 1, J = 1, ... , Lev
|        |__ SmB = 0 ; SmL = 1
|    Level = Lev
|    Go to S
|
| I: Lev = Level + 1                          INTERPOLATION
|    If Lev > MaxLev Then *Return*
|    If Lev > LevelM
|      Then Generate A(Lev) and f(Lev)
|        |  LevelM = Lev
|        |  Work(Lev) = False ; Cycle(Lev) = 1
|        |__ If Lev < MaxLev Then CycleM(Lev) = CycleC
|    If Work(Lev) = True
|      Then u(Lev) = u(Lev) + Bilinear(u(Level))
|      Else u(Lev) = LIM(u(Level))
|    Level = Lev
|___ Go to S
```

The algorithm has three computational sections. The first is <u>smoothing</u> (or direct solves on the coarsest level), the second is <u>residual correction</u>, and the last section is <u>interpolation</u>. In the latter section, LIM is an $O(h^4)$ interpolation scheme which will be defined in Section 6.2. Note that this is a more general multi-grid algorithm than either k-level scheme (Algorithms 2.4 and 2.11) defined in Chapter 2.

Brandt and his students have formulated a collection of multi-grid algorithms which adaptively choose the number of smoothing iterations on each level and subsequently decide whether to solve a residual correction problem, interpolate onto the next finer grid, or stop [14, 15, 13, 18, 40]. The principal advantage of these adaptive methods is that with good heuristics, the optimal number of correction iterations is done (too many and CPU time is wasted, too few and the solution is poor). However, fixed algorithms have three distinct advantages over these adaptive methods. First, no heuristics are needed to determine when to move from one grid to the next (and no potentially costly computation is needed to evaluate the heuristics). Since stopping criteria are unnecessary (fixed methods are really direct methods), we can precompute the operation count and choose which level scheme uses the fewest number of operations. Finally, convergence theorems exist for the fixed algorithms (see Chapters 2 - 5 and [9, 28, 29, 30, 37, 38, 49]).

We can redefine Algorithms 1.1, 1.2, 2.4, and 2.11 using the algorithm in Figure 6-1:

<u>Definition 6.1</u>: Algorithm 1.1 is defined by

KLevel(2, 1, 1, 0, m, m, s, false, 1),

where s is true if the coarse grid problem is solved by smoothing iterations and false if it is solved directly. Algorithm 1.2 is defined by

KLevel(2, 1, p, m, m, m, true, false, 2).

Algorithm 2.4 is defined by

KLevel(k, p, p, n, n+m, m, false, false, k),

where $k \geq 1$ is the number of levels used. Finally, Algorithm 2.11 is defined by

KLevel(k, p, p, n, n+m, m, false, false, 1).

We are particularly interested in four distinct k-level variants of the algorithm in Figure 6-1. We refer to them as schemes R(a), R(b), I and H. The first two are similar to the recursive multi-grid cases (a) and (b) described after Algorithm 2.4:

<u>Definition 6.2</u>: We define <u>scheme</u> <u>R</u>(<u>a</u>) by

KLevel(k, p, p, m, m, 0, false, false, 1)

and <u>scheme</u> <u>R</u>(<u>b</u>) by

KLevel(k, p, p, 0, m, m, false, false, 1).

Scheme R(a) has been analyzed by Astrakhantsev [4], Bank and Dupont [9], Hackbusch [28, 29, 30], Nicolaides [37, 38], and Van Rosendale [49] for finite-element discretizations of various elliptic boundary-value problems. Federenko [22, 23] and Bakhvalov [8] analyzed this case for finite-difference discretizations. Brandt [15] analyzed (nonrigorously) scheme R(b) for finite-difference discretizations using local mode analysis.

We represent a two-iteration four-level example of the scheme R(a) by

```
Ω8h  ds   ds       ds   ds           ds   ds       ds   ds
       \L/  \b    /  \L/  \b        /  \b/  \b    /  \b/  \b
Ω4h     m    0   m    m    0       m    m    0   m    m    0
              \L/           \b    /           \b/           \b
Ω2h            m             0   m             m             0
                              \L/                             \b
Ωh                             m                               0
```

ds	=	direct solve
m	=	m smoothing iterations
0	=	no smoothing iterations
L	=	LIM interpolation
b	=	bilinear interpolation

Where the use of LIM is indicated in the schematic above, standard finite-element implementations use the equivalent of bilinear interpolation [10, 27, 41]. Scheme R(a) always performs smoothing iterations after residual transfers to coarser grids, but not always after interpolation.

A two-iteration four-level example of scheme R(b) can be represented as:

```
Ω8h  ds   ds       ds   ds           ds   ds       ds   ds
       \L/  \b    /  \L/  \b        /  \L/  \b    /  \L/  \b
Ω4h     m    m   0    m    m       0    m    m   0    m    m
              \L/           \b    /           \L/           \b
Ω2h            m             m   0             m             m
                              \L/                             \b
Ωh                             m                               m
```

ds	=	direct solve
m	=	m smoothing iterations
0	=	no smoothing iterations
L	=	LIM interpolation
b	=	bilinear interpolation

As can be seen from the example of scheme R(b) above, smoothing iterations are always performed after interpolation, but never after residual transfers to coarser grids.

The reason for comparing schemes R(a) and R(b) is historical. At one time, we viewed interpolation as disruptive, so we believed the new error components (introduced by interpolation) should be smoothed. The analysis in Sections 4.3 and 5.3 and Theorem 2.10 shows when this impression is false. Also, residual transfer can be done in a smoothing manner, so smoothing iterations are redundant. If no smoothing iterations are performed on a grid before the first residual correction and the initial guess is zero, then the residual is just the right-hand side. This allows us to avoid computing residuals before the first correction iteration.

The last two particular schemes we consider are iterative multigrid algorithms rather than recursive:

Definition 6.3: We define scheme I by

KLevel(k, 1, p, M, 1, 1, false, false, 1)

and scheme H by

KLevel(k, p, p, 0, M, M, false, true, 1).

Scheme I, which was brought to our attention by Brandt [14], is Federenko's algorithm (Algorithm 1.2) iteratively extended to k-levels. Scheme I performs smoothing iterations whenever computation changes grids. A two-iteration four-level example of the scheme I can be represented as:

```
Ω^8h  ds           ds
        \L        /  \b
Ω^4h       1       m    1
          \L    /      \b
Ω^2h       1   m        1
            \L/          \b
Ω^h          m            1

      ds = direct solve
      m  = m smoothing iterations
      1  = one smoothing iteration
      L  = LIM interpolation
      b  = bilinear interpolation
```

Scheme H is a hybrid combination of p-1 multi-grid iterations of scheme R(b) on level k followed by one half iteration of scheme I. As we will see in Chapter 7, scheme H is almost as accurate as scheme R(b), but requires less computation. A two-iteration four-level example of the scheme H can be represented as:

```
Ω^8h  ds   ds       ds   ds          ds
        \L/  \b    /  \L/  \b        /  \L
Ω^4h     m    m   0    m    m       0    1
               \L/           \b    /      \L
Ω^2h            m             m   0        1
                               \L/          \b
Ω^h                             m            1

              ds = direct solve
              m  = m smoothing iterations
              0  = no smoothing iterations
              1  = one smoothing iteration
              L  = LIM interpolation
              b  = bilinear interpolation
```

6.2 Implementation

In this section, we provide details of some of the implementation issues raised in the last section. We define LIM interpolation and residual projection. We also discuss why injection is not sufficient for good convergence. Finally, we discuss how our FORTRAN multi-grid subroutine library is assembled.

We use a fourth order interpolation procedure whenever an initial guess is interpolated onto a grid. We define such a procedure to interpolate u^{2h} onto Ω^h by combining one-dimensional cubic interpolation and what is called the Local Inversion Method, or LIM [31]. Specifically, we do the following:

1. Use cubic interpolation to obtain the solution at the midpoints of the "lines" of the coarse grid:

```
*---X---*
|   .   |
|   .   |
|   .   |      * = coarse grid points
X.......X
|   .   |      X = fine grid points where we are
|   .   |          interpolating the solution
|   .   |
*---X---*
```

2. Use the difference equation to solve for the interpolant at the midpoints of the "boxes" of the coarse grid:

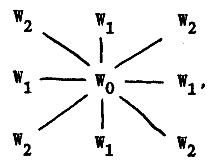
```
*---X---*
|   .   |      * = coarse grid points
|   .   |
|   .   |      X = fine grid points at midpoints
X...+...X          of coarse grid lines
|   .   |
|   .   |      + = fine grid point at midpoint
|   .   |          of coarse grid box
*---X---*
```

If $(x_i, y_j) \in \Omega^h$ corresponds to the midpoint of the box in the diagram above, then

$$u^h_{ij} = [\, f^h_{ij} - L^h_{ij}u^h_{i-1,j} - R^h_{ij}u^h_{i+1,j} - B^h_{ij}u^h_{i,j-1} - T^h_{ij}u^h_{i,j+1} \,] \,/\, D^h_{ij} + O(h^4).$$

We found that by using fourth order interpolation to make an initial guess, coupled with bilinear interpolation for corrections, one or two of the residual correction iterations were eliminated as compared to using only bilinear interpolation was used.

In Algorithm 1.2, the correction iterations are done by first computing residuals on Ω^h and then injecting them (multiplied by four) into Ω^{2h} at the points that are coincident between the two grids. In order to include the information stored in the residuals at the points not coincident between Ω^h and Ω^{2h}, we project <u>weighted</u> <u>residuals</u>. Weighting also smooths the residuals. Graphically, the weighting is given by the stencil



where we use W_0 at points coincident between Ω^h and Ω^{2h}, W_1 at the other points in the five-point discrete operator (on Ω^h), and W_2 at the four remaining points. If

$$r^h = f^h - A^h u^h,$$

then define

$$f^{2h}_{ij} = W_0\, r^h_{2i-1,2j-1} + W_1\, \{ r^h_{2i-2,2j-1} + r^h_{2i-1,2j-2} + r^h_{2i-1,2j} + r^h_{2i,2j-1} \} + W_2\, \{ r^h_{2i-2,2j-2} + r^h_{2i-2,2j} + r^h_{2i,2j-2} + r^h_{2i,2j} \}.$$

We explored the use of three different sets of weights, two standard tensor-product rules and one nonstandard rule:

	W_0	W_1	W_2
RW1	$\frac{64}{36}$	$\frac{16}{36}$	$\frac{4}{36}$
RW2	$\frac{16}{16}$	$\frac{8}{16}$	$\frac{4}{16}$
RW3	$\frac{208}{72}$	$\frac{16}{72}$	$\frac{4}{72}$

For each set of weights, W_0, W_1, W_2,

$$W_0 + 4 (W_1 + W_2) = 4$$

instead of the customary one because we assume that f^{2h} has a factor of $(2h)^2$ in it instead of h^2 (see (6.2)). We found that weighting RW1 works best when the first order terms in (6.1) are not dominant. When they are dominant, weighting RW2 works best. When (6.1) is the Laplacian or the residuals are very smooth, weighting RW3 works better than either RW1 or RW2.

We found that strict injection of residuals sometimes has an unpleasant side effect: the first correction to the approximation to u^h decreases its accuracy. The second correction restores the accuracy of the approximation to u^h to what it was before the first correction was applied. Thereafter, corrections improve the accuracy. Further, we found that this phenomenon disappears when either residual weighting is used or smoothing iterations are performed before injecting

residuals from Ω^{2h} into Ω^{4h}. We prefer weighting since it costs less than a smoothing iteration.

Our FORTRAN code is organized into a number of modules. There is an interactive driver subroutine which sets parameters used by the multi-grid subroutines, calls the library subroutines to solve a problem, and translates error codes into English. Since the driver is interactive, parameters and multi-grid algorithms can easily be changed. The driver is not needed if we are willing to set up the parameters the subroutine library requires.

We specify the differential equation using subprograms. Since the coefficient matrices are large and sparse, it is inefficient to store them in a dense manner. We have employed standard techniques similar to, but not identical to, those in the Yale Sparse Matrix Package [20, 21].

The computation is split into five modules for flexibility (e.g., a pre-conditioned conjugate gradient or SOR procedure could be substituted for the Gauss-Seidel relaxation subroutine below). Computation on the coarsest grid is treated as a special case. This module is highly dependent on the direct solver employed. We use the nonsymmetric solvers NDRV and TDRV and the symmetric ordering subroutine ODRV in the Yale Sparse Matrix Package [20, 21]. The ordering subroutine ODRV produces a minimum degree ordering based on the upper triangular part of A^h. NDRV stores the matrix factorization while TDRV does not. If sufficient memory is available, NDRV is computationally more attractive than TDRV since with it the coefficient matrix for the coarsest grid does not have to be re-factored for each direct solve.

The flow of computation on the finer grids follows the algorithm in Figure 6-1. When making a right-hand side for the next coarser grid, residuals are computed in one subroutine and weighted in another. Interpolation is performed in a subroutine, whether we want a first approximation to a solution or are adding a correction to an already existing approximate solution. Each smoothing iteration is one Gauss-Seidel relaxation sweep.

6.3 Complexity

In this section, we derive asymptotic multiply counts for three subclasses of the class of algorithms defined in Figure 6-1. If N is the number for unknowns in the finest grid Ω^h, we show that for certain choices of parameters the algorithms in each subclass require O(N) operations (asymptotically). In particular, the analysis applies to the schemes R(a), R(b), I and H, which we defined in Section 6.1. Then we show that the cost of assembling all of the coefficient matrices is only one-third higher than the cost of assembling only the coefficent matrix for the finest grid. Finally, we show that the storage requirements for the subroutine library discussed in Section 6.2 are proportional to the number of unknowns in the finest grid.

Table 6-1 contains the cost of each phase of the general multi-grid algorithm as well as the specific costs for the subroutine library described in Section 6.2. Consider the class of algorithms

Table 6-1: Generalized Multi-Grid Algorithm Costs

Operations	Value of constant in subroutine library	for each
C_1N	5.00	smoothing iteration or residual computation
C_2N	0.75	residual weighting
C_3N	2.25	LIM interpolation
C_4N	0.75	bilinear interpolation
C_5N	1.50	difference between LIM and bilinear interpolation

$$\mathrm{KLevel}(K,\ p,\ p,\ R_0,\ R_L,\ R_N,\ \cdot,\ \cdot,\ 1)$$

with finest grid Ω^h. Recall from Figure 6-1 that R_0 is the number of smoothing iterations before the first correction, R_L is the number of smoothing iterations after the last correction, and R_N is the number of smoothing iterations otherwise. Schemes R(a) and R(b) fall into this class. The cost of doing smoothing iterations on Ω^h, residual projection from Ω^h, and interpolation to Ω^h while doing p iterations of a k-level scheme is given by

$$\begin{aligned}\mathrm{Mults}(N,\ p,\ R_0,\ R_L,\ R_N) &= [C_3 + \{(p-1)(R_N+1) + R_0 + R_L\}C_1 \\ &\quad + (p-1)\{C_2 + C_4\}]N \qquad (6.3)\\ &= C_6N.\end{aligned}$$

The cost of doing these operations on coarser grids is not included in (6.3).

We begin by giving a work estimate. All logarithms are base four in the analysis. Define

$$\mathrm{Sign}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ +1 & \text{if } x > 0 \end{cases} .$$

<u>Theorem 6.4</u>: Let R_0, R_L, and R_N be fixed nonnegative integers. The total number of multiplies T for any algorithm in the class

KLevel(k, p, p, R_0, R_L, R_N, false, false, 1)

approaches the following limits asymptotically (in k):

(a) $p = 1$: $T \to \frac{4}{3}[C_3 + R_LC_1]N$

(b) $p = 2$: $T \to 2[C_3 + \{R_N + 1 + \frac{1}{3}R_0 + R_L\}C_1 + C_2 + C_4$
$\quad - \frac{1}{3}\mathrm{Sign}(R_0)C_5]N$

(c) $p = 3$: $T \to 4[C_3 + \{2(R_N + 1) + \frac{2}{3}R_0 + R_L\}C_1 + 2\{C_2 + C_4\}$
$\quad - \frac{2}{3}\mathrm{Sign}(R_0)C_5]N$

(d) $p = 4$: $T \to [C_3 + \{3(R_N + 1) + R_0 + R_L\}C_1$
$\quad + 3\{C_2 + C_4\} - \mathrm{Sign}(R_0)C_5]N\log N$
$\quad + \frac{4}{3}[\mathrm{Sign}(R_0)C_5 - R_0C_1]N$

(e) $p \to \infty$: $T \to [C_6 - \mathrm{Sign}(R_0)C_5]N^{\log p} + \frac{4}{3}\mathrm{Sign}(R_0)C_5N$.

<u>Proof</u>: Let N be the number of unknowns on level k. For each multi-grid iteration of the recursive k-level schemes we solve p (k-1)-level problems. Level k-1 has approximately N/4 unknowns, so the cost of solving all of the (k-1)-level problems is given by

$$p\ \mathrm{Mults}(N/4,\ p,\ R_0,\ R_L,\ R_N) - \frac{N}{4}[R_0C_1 + \mathrm{Sign}(R_0)C_5(p - 1)]. \qquad (6.4)$$

The term

$$\frac{N}{4}[R_0C_1 + \mathrm{Sign}(R_0)C_5(p - 1)]$$

in (6.4) is a correction. On the first (k-1)-level problem we are solving for the solution, so there are no smoothing iterations involving R_0. Since in the definition of Mults(·) R_0 is assumed to always play a role, the extra multiplies counted must be subtracted. For the remainder of the (k-1)-level correction recursion problems, when $R_0 > 0$, LIM interpolation is not used. We can inductively deduce that the total cost of a k-level scheme is

$$\begin{aligned} &\mathrm{TotMults}(k,\ N,\ p,\ R_0,\ R_L,\ R_N) \\ &\quad = \sum_{i=0}^{k-1} [p^i\ \mathrm{Mults}(4^{-i}N,\ p,\ R_0,\ R_L,\ R_N) - 4^{-i}R_0C_1N \\ &\qquad\qquad - \mathrm{Sign}(R_0)\ C_5\ \{(p/4)^i - 4^{-i}\}N] \\ &\quad = C_6N\sum_{i=0}^{k-1}(p/4)^i - R_0C_1N\sum_{i=0}^{k-1}4^{-i} \\ &\qquad\qquad - \mathrm{Sign}(R_0)C_5N\sum_{i=0}^{k-1}\{(p/4)^i - 4^{-i}\}. \end{aligned} \qquad (6.5)$$

Asymptotically,

$k = \log N,$

so

$$\sum_{i=0}^{k-1} (p/4)^i \rightarrow \begin{cases} (1 - (p/4)^{-1})^{-1}, & \text{if } p < 4 \\ \log N, & \text{if } p = 4 \; . \\ \dfrac{(p/4)^{1+\log N} - 1}{(p/4) - 1}, & \text{if } p \geq 5 \end{cases} \qquad (6.6)$$

As $p \rightarrow \infty$ in (6.6),

$$\frac{(p/4)^{1+\log N} - 1}{(p/4) - 1} \rightarrow (p/4)^{\log N} = N^{(\log p)-1}. \qquad (6.7)$$

Substitution of (6.6) and (6.7) into (6.5) yields the desired results.

QED

Substituting the subroutine library's constants from Table 6-1 into Theorem 6.4 gives us

Corollary 6.5: For the subroutine library described in Section 6.2, we have:

(a) $p = 1$: $T \rightarrow [3 + 5R_L]N$

(b) $p = 2$: $T \rightarrow [17\frac{1}{2} + 10\{R_N + \frac{1}{3}R_0 + R_L\} - \text{Sign}(R_0)]N$

(c) $p = 3$: $T \rightarrow [51 + 20\{2R_N + \frac{2}{3}R_0 + R_L\} - 4\text{Sign}(R_0)]N$

(d) $p = 4$: $T \rightarrow [21\frac{3}{4} + 5\{3R_N + R_0 + R_L\} - \frac{3}{2}\text{Sign}(R_0)]N\log N$
$+ [2\text{Sign}(R_0) - 6\frac{2}{3}R_0]N.$

We can use Corollary 6.5 to compute the number of multiplies used by the two recursive schemes singled out in Section 6.1. Substituting $R_0 = R_N = m$ and $R_L = 0$ into Corollary 6.5 gives us

Corollary 6.6: For the subroutine library, the total number of multiplies for the scheme R(a) is asymptotically (in k):

(a) $p = 1$: $T \rightarrow 3N$

(b) $p = 2$: $T \rightarrow [16\frac{1}{2} + 13\frac{1}{3}m]N$

(c) $p = 3$: $T \rightarrow [47 + 53\frac{1}{3}m]N$

(d) $p = 4$: $T \rightarrow [20\frac{1}{4} + 20m]N\log N + [2 - 6\frac{2}{3}m]N.$

Substituting $R_0 = 0$ and $R_N = R_L = m$ into Corollary 6.5 gives us

Corollary 6.7: For the subroutine library, the total number of multiplies for scheme R(b) is asymptotically (in k):

(a) $p = 1$: $T \rightarrow [3 + 5m]N$

(b) $p = 2$: $T \rightarrow [17\frac{1}{2} + 20m]N$

(c) $p = 3$: $T \to [51 + 60m]N$

(d) $p = 4$: $T \to [21\frac{3}{4} + 20m]N\log N$.

Iterative multi-grid schemes do not fall into the class of algorithms that Theorem 6.4 analyzes. For a particular subclass of iterative schemes we have

Theorem 6.8: If we let R_0, R_L, and R_N be fixed nonnegative integers, then the total number of multiplies T for any algorithm in the class

KLevel(k, 1, p, R_0, R_L, R_N, false, false, 1)

approaches the following limit asymptotically (in k):

$$T \to \frac{4}{3}[(C_1 + C_3) + (p - 1)\{(R_0 + 2)C_1 + C_2 + C_4\}]N.$$

Proof: For scheme I,

$$T = [(C_1 + C_3) + (p - 1)\{(R_0 + 2)C_1 + C_2 + C_4\}]N \sum_{i=0}^{k-1} 4^{-i}$$

$$\to \frac{4}{3}[(C_1 + C_3) + (p - 1)\{(R_0 + 2)C_1 + C_2 + C_4\}]N.$$

QED

Let $R_0 = R_N = m$ and $R_L = 1$. Substituting the particular values of p into Corollary 6.8 gives us

Corollary 6.9: For the subroutine library, the total number of multiplies for scheme I is asymptotically (in k):

(a) $p = 1$: $T \to 9\frac{2}{3}N$

(b) $p = 2$: $T \to [25 + 6\frac{2}{3}m]N$

(c) $p = 3$: $T \to [40\frac{1}{3} + 13\frac{1}{3}m]N$

(d) $p = 4$: $T \to [55\frac{2}{3} + 20m]N$

The last theorem combines the multiply count of scheme R(b) with that of scheme I.

Theorem 6.10: For the subroutine library, the total number of multiplies for scheme H is asymptotically (in k):

(a) $p = 2$: $T \to [18\frac{1}{8} + 10m]N$

(b) $p = 3$: $T \to [45\frac{3}{4} + 40m]N$

(c) $p = 4$: $T \to [16\frac{5}{16} + 15m]N\log N + [26\frac{3}{4} + 15m]N$.

Proof: For scheme H, $R_0 = 0$ and $R_N = R_L = m$. For $p > 1$,

$$T = (p-1)\text{TotMults}(K-1,\ N/4,\ p,\ 0,\ m,\ m)$$

$$+ \text{Mults}(N,\ p,\ 0,\ 1,\ m) + [C_1 + C_2 + C_3]N \sum_{i=1}^{k-1} 4^{-i}$$

$$= (p-1)\text{TotMults}(K-1,\ N/4,\ p,\ 0,\ m,\ m)$$

$$+ \text{Mults}(N,\ p,\ 0,\ 1,\ m) + \tfrac{1}{3}[C_1 + C_2 + C_3]N.$$

Substitution of particular values of p into the above equation proves the result.

QED

We now turn to analyzing the cost of assembling the coefficient matrices associated with each grid. A coefficient matrix is generated and saved for each level. At each point in a grid, functions are evaluated to get the values of P, Q, V, W, and S for the differential equation (6.1). If N_i is the number of unknowns in the grid associated with each level i, i = 1, 2, ... , k, then we have the following bound:

Theorem 6.11: Assembling the multi-grid coefficient matrices requires one-third more function evaluations than assembling just the finest-grid coefficient matrix asymptotically in k.

Proof: On the finest grid, $5N_k$ function evaluations are required. The number of function evaluations required by multi-grid is

$$5 \sum_{i=0}^{k-1} N_i \cong 5N_k \sum_{i=0}^{k-1} 4^{-i} \to \frac{4}{3}(5N_k) \quad \text{as } k \to \infty.$$

QED

We can derive bounds for the number of multiplies needed to assemble a coefficient matrix A^h:

Proposition 6.12: Let N be the rank of A^h. Excluding the cost of the function evaluations, the cost of assembling A^h is

$$4N \le \text{Cost} \le 15N.$$

The variation in the cost in Proposition 6.12 comes from whether or not any of the terms V, W, and S in (6.1) are zero. Thus, the following bound is immediate from Theorem 6.11 and Proposition 6.12:

Corollary 6.13: Assembling the multi-grid coefficient matrices costs one-third more than assembling just the coefficient matrix for the finest grid asymptotically in k.

When the coefficient functions are expensive to evaluate, they can be evaluated once on the finest-grid and saved in tables. In this case, additional function evaluations are inexpensive table look-ups and the cost of assembling the multi-grid coefficient matrices is about the same as just assembling the finest-grid matrix. This is the technique used to assemble the right-hand sides.

Finally, we consider the amount of storage required by the subroutine library. This implementation stores every discretization as a nonsymmetric system of equations, even for self-adjoint problems. Table 6-2 contains the storage requirements of the library for all algorithms in class

Table 6-2: Generalized Multi-Grid Algorithm Storage Requirements

Locations	for
$8N_1$	Yale Sparse Matrix Package data structure
$3N_i$	approximate solution, right-hand side, and internal data structure, $i = 1, 2, \ldots, k$
$5N_i$	coefficient matrix, $i = 1, 2, \ldots, k$
N_k	residuals

KLevel(k, ·, ·, R_0, R_L, R_N, false, ·, ·).

Theorem 6.14: The total number of locations needed (excluding the memory required by the Yale Sparse Matrix Package for the factorization on the coarsest grid) is asymptotically (in k) given by

$$\text{Storage} \rightarrow 11\tfrac{2}{3}\, N_k.$$

Proof: From Table 6-2,

$$\text{Storage} = [\, 1 + 8\, \{\, 4^{1-k} + \sum_{i=0}^{k-1} 4^{-i} \,\}\,]\, N_k.$$

$$\rightarrow 11\tfrac{2}{3}\, N_k \quad \text{as } k \rightarrow \infty.$$

QED

Corollary 6.15: With the same assumptions as in Theorem 6.14, the total number of locations needed for self-adjoint problems should be given by

$$\text{Storage} \rightarrow 9N_k.$$

Proof: Since the problem is symmetric, the number of locations used by the coefficient matrix on level i is $3N_i$, $i = 1, 2, \ldots, k$. So,

$$\text{Storage} = [\, 1 + 6\, \{\, 4^{1-k} + \sum_{i=0}^{k-1} 4^{-i} \,\}\,]\, N_k,$$

and the result follows as before.

QED

CHAPTER 7

Numerical Experiments

7.1 Experiments

In this section, we present results of numerical experiments comparing the particular multi-grid schemes of Section 6.1. In these experiments, the domain Ω is the unit square $[0,1]x[0,1]$. We find that the multi-grid schemes R(a), R(b), I and H use less computer time and require less storage than sparse Gaussian elimination to achieve similar accuracy. However, the two-level schemes of Southwell and Federenko (Algorithms 1.1 and 1.2) require more computer time than sparse Gaussian elimination to achieve similar accuracy. We also investigate the relative efficiencies of multi-grid and sparse Gaussian elimination for the case in which a collection of partial differential equations with the same differential operator, but with different right-hand sides needs to be solved.

Our test problem is a variable coefficient, nonself-adjoint equation:

$$\begin{cases} -(e^{-xy}u_x)_x - (e^{xy}u_y)_y + (1/2 - y)u_x \\ \qquad + (x - 1/2)u_y - u/(1 + x + y) = f \text{ in } \Omega \\ u = 0 \text{ on } \partial\Omega, \end{cases}$$

where the right-hand side f is constructed so that the solution is $u = xe^{xy}\sin(\pi x)\sin(\pi y)$. This problem is similar to Ellpack test problem one [42].

The amount of storage reported in Table 7-1 is the number of single-precision words required for the approximate solutions, right-hand sides, coefficient matrices, and factorization of the coarsest grid matrix. The direct solve on the finest grid uses O(NlogN) storage for the five-point discretization (6.2) [45], where N is the number of points in the finest grid. In the multi-grid case, the ratio of storage to N is slightly above that predicted in Theorem 6.14. The discrepancy is caused by our discounting in Theorem 6.14 the space required to save the factorization of the coarsest grid's coefficient matrix. As can be seen in Table 7-1, most of the savings in space occurs when just two or three levels are used.

The results for the test problem are summarized in Table 7-2. The table contains data for finest grids of 17x17, 33x33, and 65x65. For each fine-grid, we report the following data for p cycles $(2 \lesssim p \lesssim 4)$ and two Gauss-Seidel relaxation sweeps per iteration of each of the particular multi-grid schemes of Section 6.1:

Table 7-1: Storage Requirements for the FORTRAN Multi-Grid Library

N	Levels 1	1	2	3	4	5
17x17	9,547	13,219	4,803	4,168		
	33.0	45.7	16.6	14.4		
33x33	41,153	66,179	20,979	15,716	15,081	
	37.8	60.8	19.3	14.4	13.8	
65x65	187,663	*	96,743	63,268	58,005	57,370
	44.4		22.9	15.0	13.7	13.6
Direct solver	TDRV	NDRV	NDRV	NDRV	NDRV	NDRV

Storage (in single-precision words) and Storage/N

*There was not enough memory available to solve this problem.

- the time (in CPU seconds) to solve the discrete system of equations, and

- the maximum absolute error (with respect to the solution of the differential equation) at the grid points of the finest grid.

The timings were performed on a lightly loaded DEC-2060 computer running TOPS-20.

From Table 7-2, we draw several conclusions for the test problem. On the finest grid all of the multi-grid schemes achieve their accuracy using much less CPU time than the direct solve. Consider the 65x65, p = 2, 5-level case in Table 7-2. Scheme H and scheme R(a) require 1.7 CPU seconds, scheme R(b) requires 2.1 seconds, while the ODRV + TDRV direct solve requires 38.8 seconds. There was not enough memory for NDRV to solve the 65x65 problem without modifying the subroutine (the

Table 7-2: Test Problem Results

NDRV + ODRV is used unless otherwise specified. All times are in DEC-2060 CPU seconds. All errors are the maximum absolute error with respect to the solution of the differential equation, evaluated at the fine grid points.

(a) 17x17 Finest Grid

Direct Solve on Finest Grid: 0.531 CPU seconds
0.522 CPU seconds (TDRV)
2.02E-03 Maximum Error

Scheme	p	Level 2	3
R(a)	1	0.084	0.027
		8.07E-03	3.94E-02
	2	0.122	0.104
		2.11E-03	3.21E-03
R(a) without using LIM	1	0.086	0.029
		5.50E-02	1.95E-01
	2	0.132	0.102
		1.14E-02	1.53E-02
	3	0.190	0.204
		2.37E-03	2.43E-03
	4	0.225	0.343
		1.86E-03	1.85E-03
R(b)	1	0.103	0.053
		7.00E-03	2.20E-02
	2	0.149	0.125
		1.99E-03	2.09E-03

Scheme	p	Level 2	3
I	1	0.103	0.026
		7.33E-03	2.81E-02
	2	0.143	0.094
		2.02E-03	4.75E-03
	3	0.190	0.153
		2.06E-03	2.54E-03
	4	0.238	0.193
		2.02E-03	2.11E-03
H	2	0.144	0.102
		2.02E-03	2.38E-03

(b) 33x33 Finest Grid

```
Direct Solve on Finest Grid:      3.902 CPU seconds      (NDRV)
                                  3.879 CPU seconds      (TDRV)
                              5.09E-04 Maximum Error
```

Scheme	p	Levels 2	Levels 3	Levels 4
R(a)	1	0.569	0.125	0.067
		2.02E-03	8.07E-03	3.94E-02
	2	0.768	0.421	0.434
		4.66E-04	5.01E-04	7.29E-04
R(a) without using LIM	1	0.566	0.117	0.065
		1.93E-02	5.50E-02	1.95E-01
	2	0.767	0.375	0.427
		1.02E-02	1.03E-02	1.05E-02
	3	0.966	0.819	1.023
		1.16E-03	1.23E-03	1.23E-03
	4	1.167	1.286	1.910
		4.75E-04	4.80E-04	4.80E-04
R(b)	1	0.650	0.223	0.168
		1.96E-03	6.88E-03	2.14E-02
	2	0.849	0.517	0.522
		4.58E-04	4.60E-04	4.63E-04
I	1	0.606	0.165	0.123
		1.98E-03	7.26E-03	2.76E-02
	2	0.800	0.388	0.343
		4.58E-04	7.09E-04	3.44E-03
	3	1.006	0.601	0.563
		5.26E-04	6.17E-04	1.18E-03
	4	1.196	0.821	0.787
		5.09E-04	5.21E-04	6.20E-04
H	2	0.801	0.418	0.402
		4.58E-04	4.99E-04	5.25E-04

(c) 65x65 Finest Grid

```
Direct Solve on Finest Grid:        not enough memory    (NDRV)
                                    38.8 CPU seconds     (TDRV)
                                 1.24E-04 Maximum Error
```

Scheme	p	Levels 2	Levels 3	Levels 4	Levels 5
R(a)	1	4.135	0.691	0.270	0.219
		5.08E-04	2.02E-03	8.07E-03	3.95E-02
	2	5.009	1.963	1.726	1.469
		1.10E-04	1.16E-04	1.21E-04	1.64E-04
	3	5.874	3.619	4.073	4.831
		1.32E-04	1.32E-04	1.32E-04	1.32E-04
R(a) without using LIM	1	4.093	0.490	0.253	0.127
		5.66E-03	1.93E-02	5.50E-02	1.95E-01
	2	5.032	1.982	1.701	1.762
		1.07E-02	1.06E-02	1.06E-02	1.06E-02
	3	5.829	3.635	3.936	4.532
		5.88E-04	6.23E-04	6.26E-04	6.26E-04
	4	6.773	5.649	7.601	10.234
		2.44E-04	2.51E-04	2.51E-04	2.51E-04
R(b)	1	4.511	1.145	0.683	0.634
		5.04E-04	1.95E-03	6.83E-03	2.12E-02
	2	5.300	2.340	2.086	2.149
		1.10E-04	1.10E-04	1.11E-04	1.11E-04
I	1	4.309	0.917	0.463	0.423
		5.06E-04	1.98E-03	7.23E-03	2.74E-02
	2	5.127	1.808	1.377	1.336
		1.10E-04	1.28E-04	4.62E-04	3.15E-03
	3	6.095	2.720	2.286	2.262
		1.32E-04	1.57E-04	2.51E-04	8.38E-04
	4	6.944	3.586	3.227	3.177
		1.27E-04	1.28E-04	1.41E-04	2.55E-04
H	2	5.142	1.949	1.619	1.664
		1.10E-04	1.18E-04	1.20E-04	1.21E-04

DEC-2060 has an addressing limitation of 256K single-precision words).

When p = 2 or 3, the multi-grid schemes actually require time proportional to the number of unknowns in the finest grid, just as predicted by Corollaries 6.6, 6.7, and 6.9 and Theorem 6.10. The multiply counts in Section 6.3 accurately predict the difference between the running times of the different schemes. The maximum number of levels minimizes the storage required. While this will not always minimize the time required, it will almost do this. Scheme H and scheme R(a) are the fastest multi-grid schemes tested which achieve accuracy comparable to a direct solve. Scheme R(b) requires approximately 25% more time than either scheme H or scheme R(b) to achieve similar accuracy. Scheme I costs more than the other particular multi-grid schemes. To be fair, this scheme is best suited to problems where a good initial guess to the finest grid solution exists.

The use of LIM to generate initial guesses instead of bilinear interpolation decreases the number of correction iterations required to achieve accuracy similar to a direct solve. This is particularly noticeable when many levels are being used. In a two-level scheme, the direct solve provides a good enough approximation to the fine-grid solution so that interpolation plays a lesser role.

Table 7-3 contains the results of timing the test problem without including the re-ordering and factorization of the coarsest grid's coefficient matrix. We draw some conclusions about solving a sequence of linear systems which have the same coefficient matrices. When

Table 7-3: Test Problem Timings Excluding Factorization Time

Finest Grid:		17x17		33x33			65x65			
Direct Solve:		0.043 CPU seconds		0.240 CPU seconds			not enough memory			
		Levels		Levels			Levels			
Scheme	p	2	3	2	3	4	2	3	4	5
R(a)	1	0.019	0.016	0.077	0.058	0.049	0.348	0.229	0.183	0.026
	2	0.066	0.092	0.280	0.358	0.415	1.259	1.468	1.640	1.782
	3	0.113	0.200	0.479	0.733	1.007	2.133	3.149	4.025	4.830
R(a) without using LIM	1	0.018	0.017	0.080	0.055	0.053	0.379	0.313	0.156	0.201
	2	0.066	0.091	0.256	0.354	0.420	0.965	1.477	1.653	1.833
	3	0.113	0.194	0.478	0.741	1.010	2.129	3.065	3.862	4.803
R(b)	1	0.038	0.036	0.161	0.153	0.160	0.708	0.641	0.570	0.631
	2	0.086	0.114	0.359	0.449	0.510	1.585	1.856	2.009	2.134
I	1	0.023	0.027	0.119	0.106	0.108	0.560	0.441	0.418	0.411
	2	0.073	0.084	0.323	0.323	0.333	1.423	1.289	1.338	1.188
	3	0.121	0.142	0.518	0.536	0.552	2.300	2.224	2.234	2.247
	4	0.170	0.195	0.717	0.752	0.776	3.159	3.141	3.141	3.175
H	2	0.077	0.092	0.283	0.354	0.394	1.408	1.447	1.566	1.640

solving systems of the form Au = f by sparse Gaussian elimination techniques, backsolves take <u>much</u> less time than re-ordering and factoring the matrix A. When the grids are small and two or more iterations of a k-level scheme are used, two levels should be used to ensure the lowest multi-grid running time.

When the computer charges are based only on CPU time, the sparse Gaussian elimination techniques are less expensive than multi-grid if sufficiently many systems are to be solved and the grids are small enough. However, recall that for $p = 2$ or 3, the work estimate for multi-grid is $O(N)$, while for a backsolve, the work estimate is $O(N\log N)$. Thus, asymptotically (in N), multi-grid is less expensive than sparse Gaussian elimination.

Even when the grids are small so that multi-grid requires more CPU time than a backsolve, multi-grid schemes are very competitive when the charge is determined by the kilo-core seconds rather than by the number of CPU seconds. Suppose the charging algorithm is

$$\text{Charge} = (\text{CPU seconds used}) \times (\text{Memory used} / 1000)^{\omega},$$

where ω is a non-negative real number. We can determine for what values of ω it is less expensive to use a multi-grid algorithm instead of a direct solve on the finest grid. This requires solving

$$[(\text{Direct solve time}) / (\text{Direct solve storage})]^{\omega} \geq (\text{Multi-grid time}) / (\text{Multi-grid storage})$$

for ω, i.e.,

$$\omega \geq \frac{\log[(\text{Multi-grid time})/(\text{Direct solve time})]}{\log[(\text{Direct solve storage})/(\text{Multi-grid storage})]}$$

Consider the case of a 33x33 finest grid in Table 7-3. Scheme H is less expensive than a direct solve whenever

$$\omega \geq \begin{cases} .27 & \text{for two levels} \\ .28 & \text{for three levels.} \\ .36 & \text{for four levels} \end{cases}$$

Common values of ω are 0, .5, and 1.

Table 7-4 contains the results of solving the test problem by Gauss-Seidel and the algorithms of Southwell and Federenko (Algorithms 1.1 and 1.2). In Southwell's algorithm, we solve the coarse grid problem directly using the Yale Sparse Matrix Package [20, 21]. For Federenko's algorithm and Gauss-Seidel, we use an initial guess of zero for the fine-grid solution. Some conclusions can be drawn from Table 7-4. First, the algorithms of Federenko and Southwell appear to work equally well, but Gauss-Seidel takes a <u>very</u> long time. Second, none of these three methods are competitive timewise with a direct solve on the fine grid or any of the k-level schemes defined in Section 6.1.

7.2 Conclusions

In conclusion, we find that the k-level schemes use substantially less computer time and storage than sparse Gaussian elimination to achieve similar accuracy. We can solve the test problems using fewer correction iterations using LIM plus bilinear interpolation and residual weighting than if we solve them using just bilinear

Table 7-4: Test Problem Results for the Methods of Federenko, Southwell, and Gauss-Seidel

Southwell's method uses NDRV. All times are in DEC-2060 CPU seconds. All errors are the maximum absolute error with respect to the solution of the differential equation, evaluated at the points of the fine grid.

Fine Grid	Federenko's Method				Southwell's Method			Gauss-Seidel		
	p	m	Error	Time	m	Error	Time	m	Error	Time
17x17	2	15	3.01E-02	0.348	2	7.05E-03	0.127	100	1.56E-02	0.985
	3	15	1.91E-03	0.540	10	5.41E-03	0.186	150	2.13E-03	1.445
					50	2.54E-03	0.577			
33x33	2	25	1.73E-01	2.388	2	1.96E-03	0.702	100	2.75E-01	4.132
	4	25	1.67E-02	5.095	100	9.01E-04	4.694	300	4.03E-02	12.463
	6	25	1.44E-03	7.767	200	6.30E-04	8.746	500	5.88E-03	20.733
	8	25	3.96E-04	10.409				700	7.45E-04	29.155

interpolation. The two-level schemes of Southwell and Federenko require more computer time than sparse Gaussian elimination to achieve similar accuracy, but less than Gauss-Seidel.

When a collection of partial differential equations with the same differential operator, but different right-hand sides is to be solved, sparse Gaussian elimination is sometimes less expensive than multi-grid. When the computer charging algorithm is based on CPU time instead of kilo-core seconds, sparse Gaussian elimination is less expensive if sufficiently many systems are solved, there is sufficient memory available to save the factorization of the coefficent matrix, and the grids are small. When the computer charging algorithm is based on kilo-core seconds, we have shown when multi-grid is less expensive to use than sparse Gaussian elimination. Otherwise, multi-grid is less expensive than sparse Gaussian elimination.

In this chapter, we have shown empirically that a variety of multi-grid schemes can be implemented which lead to optimal order solvers for self-adjoint problems. Whether scheme R(b), scheme H, or scheme R(a) is used makes little difference since they all behave in a similar manner. We have also shown that multi-grid methods compare quite favorably with sparse Gaussian elimination, even for nonself-adjoint problems. However, a warning about multi-grid methods is warranted: implementing these methods for general problems can be extremely time consuming.

Bibliography

[1] S. Agmon, Lectures on Elliptic Boundary Value Problems, Van Nostrand Reinhold, New York, 1965.

[2] A. Aho, J. Hopcroft, and J. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA, 1974, pp. 64-65.

[3] R. E. Alcouffe, A. Brandt, J. E. Dendy, Jr., J. W. Painter, The multi-grid methods for the diffusion equation with strongly discontinuous coefficients, SIAM J. on Sci. and Stat. Comp., 2 (1981), pp. 430-454.

[4] G. P. Astrakhantsev, An iterative method of solving elliptic net problems, Z. Vycisl. Mat. i. Mat. Fiz., 11 (1971), pp. 439-448.

[5] O. Axelsson and I. Gustafsson, A Combination of Preconditioning and Multigrid Methods, Technical Report 8120, Universty of Nijmegen, 1981.

[6] O. Axelsson and I. Gustafsson, A modified upwind scheme for convective transport equations and the use of a conjugate gradient method for the solution of non-symmetric systems of equations, J. Inst. Maths Applics, 23 (1979), pp. 321-337.

[7] I. Babuska and A. K. Aziz, Survey lectures on the mathematical foundations of the finite element method, A. K. Aziz, ed., The Mathematical Foundations of the Finite Element Method with Applications to Partial Differential Equations, Academic Press, New York, 1972, pp. 3-359.

[8] N. S. Bakhvalov, On the convergence of a relaxation method under natural constraints on an elliptic operator, Z. Vycisl. Mat. i. Mat. Fiz., 6 (1966), pp. 861-883.

[9] R. E. Bank and T. Dupont, An optimal order process for solving elliptic finite element equations, Math. Comp., 36 (1981), pp. 35-51.

[10] R. E. Bank and A. Sherman, PLTMG User's Guide, Technical Report 152, Center for Numerical Analysis, University of Texas, 1979.

[11] J. H. Bramble and S. R. Hilbert, Bounds for a class of linear functionals with applications to Hermite interpolation, Numer. Math., 16 (1971), pp. 362-369.

[12] J. H. Bramble and M. Zlamal, Triangular elements in the finite element method, Math. Comp., 24 (1970), pp. 809-821.

[13] A. Brandt and N. Dinar, Multigrid solutions to elliptic flow problems, S. Parter, ed., Numerical Methods for Partial Differential Equations, Academic Press, New York, 1979, pp. 53-147.

[14] A. Brandt, Lecture notes of the ICASE Workshop on multi-grid methods, ICASE, NASA Langley Research Center, 1978.

[15] A. Brandt, Multi-level adaptive solutions to boundary-value problems, Math. Comp., 31 (1977), pp. 333-390.

[16] P. G. Ciarlet, The Finite Element Method for Elliptic Problems, North-Holland, Amsterdam, 1978.

[17] J. E. Dendy, Jr. and J. M. Hyman, Multi-grid and ICCG for problems with interfaces, M. H. Schultz, ed., Elliptic Problem Solvers, Academic Press, New York, 1981, pp. 247-253.

[18] N. Dinar, Fast Methods for the Numerical Solution of Boundary Value Problems, PhD Thesis, Weizmann Institue of Science, 1979.

[19] C. Douglas, The Effect of Interpolation on the Rate of Convergence in Two-Level Algorithms for Elliptic Partial Differential Equations, in preparation.

[20] S. C. Eisenstat, M. C. Gursky, M. H. Schultz, and A. H. Sherman, Yale Sparse Matrix Package: I. The Symmetric Codes, Technical Report 112, Department of Computer Science, Yale University, 1977.

[21] S. C. Eisenstat, M. C. Gursky, M. H. Schultz, and A. H. Sherman, Yale Sparse Matrix Package: II. The Nonsymmetric Codes, Technical Report 114, Department of Computer Science, Yale University, 1977.

[22] R. P. Federenko, A relaxation method for solving elliptic difference equations, Z. Vycisl. Mat. i. Mat. Fiz., 1 (1961), pp. 922-927.

[23] R. P. Federenko, The speed of convergence of one iteration process, Z. Vycisl. Mat. i. Mat. Fiz., 4 (1964), pp. 559-563.

[24] H. Foerster, K. Stuben, and U. Trottenberg, Non-standard multigrid techniques using checkered relaxation and intermediate grids, M. H. Schultz, ed., Elliptic Problem Solvers, Academic Press, New York, 1981, pp. 285-300.

[25] G. Forsythe and C. Moler, Computer Solution of Linear Algebraic Systems, Prentice-Hall, Englewood Cliffs, 1967.

[26] P. Grisvard, Behaviour of the solutions of an elliptic boundary value problem in a polygonal or polyhedral domain, B. Hubbard, ed., Numerical Solution of Partial Differentail Equations - III, Academic Press, New York, 1976, pp. 207-274.

[27] W. Hackbusch, A Multi-Grid Method Applied to a Boundary Value Problem with Variable Coefficients in a Rectangle, Technical Report 77-17, Mathematisches Institut, Universitat zu Koln, 1977.

[28] W. Hackbusch, Convergence of multi-grid iterations applied to difference equations, Math. Comp., 34 (1980), pp. 425-440.

[29] W. Hackbusch, On the convergence of a multi-grid iteration applied to finite element equations, Technical Report 77-8, Mathematisches Institut, Universitat zu Koln, 1977.

[30] W. Hackbusch, On the convergence of multi-grid iterations, Beit. zur Num. Math., 9 (1981), pp. 213-239.

[31] J. H. Hyman, Mesh refinement and local inversion of elliptic differential equations, J. of Comp. Phys., 23 (1977), pp. 124-134.

[32] E. Isaacson and H. Keller, Analysis of Numerical Methods, John Wiley and Sons, New York, 1966.

[33] P. Jamet, Estimations d'erreur pour des elements finis droits presque degeneres, Rev. Francaise Automat. Informat. Recherche Operationelle, Ser. Rouge, 10 (1976), pp. 43-61.

[34] R. B. Kellog, Higher order singularites for interface problems, A. K. Aziz, ed., The Mathematical Foundations of the Finite Element Method with Applications to Partial Differential Equations, Academic Press, New York, 1972, pp. 589-602.

[35] R. Kettler and J. A. Meijerink, A Multigrid Method and a Combined Multigrid-Conjugate Gradient Method for Elliptic Problems with Strongly Discontinuous Coeeficients in General Domains, Technical Report 604, Shell Oil Company, 1981.

[36] J. L. Lions and E. Magenes, Problemes aux limites non homogenes et applications, I, Dunod, Paris, 1968.

[37] R. A. Nicolaides, On multigrid convergence in the indefinite case, Math. Comp., 32 (1978), pp. 1082-1086.

[38] R. A. Nicolaides, On the 1^2 convergence of an algorithm for solving finite element equations, Math. Comp., 31 (1977), pp. 892-906.

[39] R. A. Nicolaides, On the multiple grid and related techniques for solving discrete elliptic systems, J. of Comp. Phys., 19 (1975), pp. 418-431.

[40] D. Ophir, Language for Processes of Numerical Solutions to Differential Equations, PhD Thesis, Weizmann Institue of Science, 1979.

[41] T. C. Poling, Numerical Experiments with the Multi-Grid Method, Master's Thesis, College of William and Mary, 1978.

[42] J. R. Rice, EllPack 77 User's Guide, Technical Report CSD-TR 226, Purdue University, 1978.

[43] M. H. Schultz, Spline Analysis, Prentice-Hall, Englewood Cliffs, 1973.

[44] R. Scott, Interpolated boundary conditions on the finite element method, SIAM J. of Numer. Anal., 12 (1975), pp. 404-427.

[45] A. H. Sherman, On the Efficient Solution of Sparse Systems of Linear and Nonlinear Equations, PhD Thesis, Yale University, 1975.

[46] R. V. Southwell, Relaxation Methods in Engineering Science, Oxford University Press, Oxford, 1940.

[47] E. L. Stiefel, Uber einige methoden der relaxationsrechnung, Z. Angew. Math. Phys., 3 (1952), pp. 1-33.

[48] G. Strang and G. J. Fix, An Analysis of the Finite Element Method, Prentice-Hall, New York, 1973.

[49] J. R. Van Rosendale, Rapid Solution of Finite Element Equations on Locally Refined Grids by Multi-Level Methods, PhD Thesis, University of Illinois, 1980.

[50] R. S. Varga, Matrix Iterative Analysis, Prentice-Hall, New York, 1962, pp. 181-186.