ABSTRACT OF DISSERTATION

Karen Munsell Walters

The Graduate School

University of Kentucky

2003

Variable Order Cascadic Multigrid

ABSTRACT OF DISSERTATION

A dissertation submitted in partial fulfillment of the
requirements for the degree of Doctor of Philosophy
in the College of Arts and Sciences at the University of Kentucky

By

Karen Munsell Walters

Lexington, Kentucky

Co-Directors: Dr. Craig C. Douglas, Department of Computer Science, and Dr. Carl Lee, Department of Mathematics

Lexington, Kentucky

2003

ABSTRACT OF DISSERTATION

Variable Order Cascadic Multigrid

Cascadic multigrid algorithms have been used in a variety of applications. Cascadic multigrid differs from traditional one way multigrid methods in that the problem is over solved on the coarse grids in order to reduce the required work on the finest grids. Since there is a geometric growth in the number of unknowns as the grids get finer, useful work done on the coarse grids has a significant impact on the work estimate for the entire process. Instead of over solving the problem on the coarser grids, a new method is developed to use a higher order discretization and a higher order operator based interpolation on these grids. The results is an inexpensive, high order cascadic multigrid algorithm.

KEYWORDS: Cascadic Multigrid, Initial and Boundary Value Problems, Partial Differential Equations, Numerical Linear Algebra, High Performance Computing

Karen Munsell Walters

Date

Variable Order Cascadic Multigrid

By

Karen Munsell Walters

Co-Director of Dissertation

Co-Director of Dissertation

Director of Graduate Studies

Date

RULES FOR THE USE OF DISSERTATIONS

Unpublished dissertations submitted for the Doctor's degree and deposited in the University of Kentucky Library are as a rule open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but quotations or summaries of parts may be published only with the permission of the author, and with the usual scholarly acknowledgments.

Extensive copying or publication of the dissertation in whole or in part requires also the consent of the Dean of the Graduate School of the University of Kentucky.

A library that borrows this dissertation for use by its patrons is expected to secure the signature of each user.

Name	Date

DISSERTATION

Karen Munsell Walters

The Graduate School

University of Kentucky

2003

Variable Order Cascadic Multigrid

DISSERTATION

A dissertation submitted in partial fulfillment of the
requirements of the degree of Doctor of Philosophy
in the College of Arts and Sciences at the University of Kentucky

By

Karen Munsell Walters

Lexington, Kentucky

Co-Directors: Dr. Craig C. Douglas, Department of Computer Science, and Dr. Carl Lee, Department of Mathematics

Lexington, Kentucky

2003

Table of Contents

List of Figures

Chapter 1

Introduction

Multigrid is an approach to computational problem solving originally developed to solve boundary value problems arising in physical applications. In the multigrid algorithm, the problem is discretized on a series of grids. The solution on each grid is an approximation to the continuous problem and is used as an initial guess for the next finer grid. At certain points in the algorithm, the residual on a particular level is projected to the next coarser level in order to damp the high frequency components of the error. This is known as the correction step of the process.

The components of multigrid are summarized in Chapter 2. Section 2.1 of this chapter contains the preliminary terminology used throughout this work. Descriptions of the types of boundary value problems that can be solved using the multigrid method are given in Section 2.2. A boundary value problem is a partial differential equation coupled with boundary conditions. Examples of different discretization methods used in multigrid are included in Section 2.3. The problem is solved on a particular grid using direct or iterative methods. Such methods are defined in Section 2.4. A detailed description of the various multigrid algorithms is given in Section 2.5.

Cascadic multigrid evolved from the *Cascade Principle* of Deuflhard, Leinen, and Yserentant [11], as described in Section 3.1. In the cascadic multigrid method, the problem begins on the coarsest level and is over solved on each level to provide an initial solution on the next finer level. Once a solution has been found on a particular grid, the method moves to the next finer grid without returning to the former grid, thus there is no correction. The cascadic conjugate gradient method for elliptic problems is described in Section 3.2. Subsection 3.2.1 describes the cascadic conjugate gradient method with preconditioning and Subsection 3.2.2 covers the method without the use of a preconditioner. Cascadic multigrid methods with general smoothers such as symmetric Gauss-Seidel, SSOR, or damped Jacobi (see Section 2.4 for a description of these smoothers) are given in Section 3.3. I have expanded the proofs of Bornemann and Deuflhard [3] to show more detail and have provided a proof of Lemma 1.5 [3]. An example of a problem solved using cascadic multigrid with general smoothers is Poisson's equation with the P1 conforming element (see Subsection 3.3.1). Nonconforming elements and conforming elements other than the P1 conforming element are covered in Section 3.4. A more precise statement of a lemma from [22] is given in this section. The plate bending problem is an example in which the Morley finite element is used. Section 3.5 describes the adaptive cascadic multigrid method, which is often used when the grid is nonuniform. New results are given in Section 3.6, where a cascadic multigrid algorithm is implemented in which a higher order discretization is used on the coarser levels to accelerate the convergence. On these coarser levels, a lower order method can only solve up to the truncation error, thus the higher order method allows us to over solve the problem. At some point, there is a switch to the standard (i.e., lower) method on the finer levels. One reason for switching to a lower order method on the finer grids is that it is cheaper to use than the higher order method and cost can be a significant factor as the number of grid points gets larger. Not only is the smoother more expensive, but the interpolation process is also. Another reason is that, in a number of cases, a problem may arise that involves some post processing on the finest level that requires a particular order discretization. For example, in a combustion or computational fluid dynamics application, an auxiliary problem may be solved to recover the actual solution. If a fourth order method is used, problems may arise with continuity constraints on the data. For example, if there are sharp changes along a front, the high order method could result in a Gibb's effect, which completely destroys the value of the solution. The first six sections of Chapter 3 cover the solutions of elliptic problems. Parabolic problems are examined in the final section, Section 3.7.

In Chapter 4, the results of four numerical experiments performed using the cascadic multigrid method with the switch to a lower order method are given. Section 4.1 covers the one dimensional Poisson's problem on the interval [0,1] with Dirichlet boundary conditions. In Section 4.2, two different two dimensional examples are given. The first is Poisson's equation with Dirichlet boundary conditions on the unit square. The second two dimensional problem studies the flow through a field. Section 4.3 revisits the Poisson problem in three dimensions.

Chapter 2

Definitions and Terminology

In this chapter, the terms used throughout the remaining chapters are defined. Section 2.2 describes the terminology related to the class of problems to be solved. In Section 2.3, the finite difference and finite element discretization methods are described. Section 2.4 describes the direct and iterative solvers used in the cascadic algorithms chapter (Chapter 3). A summary of notation and norm definitions is provided in the glossary.

2.1 Preliminary definitions

Boundary value problems, as described in Section 2.2, are defined on certain function spaces. The *linear space* [12] is the foundation on which other spaces are defined. A linear space is a set of vectors X paired with a set of scalars Y, such that if $x, y, z \in X$ and $a, b \in Y$, then

1. $x + y = y + x$,

2. $x + (y + z) = (x + y) + z$,

3. There exists $0 \in Y$ such that $0 \cdot x = 0$ for all $x \in X$,

4. $a(bx) = (ab)x$,

5. $(a + b)x = ax + bx$,

6. $1 \cdot x = x$, and

7. $a(x + y) = ax + ay$.

A *normed linear space* [7] is a linear space on which there is defined a real-valued *norm* $\|x\|$ for x in the space satisfying:

1. $\|x\| \geq 0$, and $\|x\| = 0$ iff $x = 0$,

2. $\|cx\| = |c| \cdot \|x\|$ for scalar c, and

3. $\|x_1 + x_2\| \leq \|x_1\| + \|x_2\|$.

The notation $C(\Omega)$ represents the linear space of continuous functions on a domain Ω, and $C^k(\Omega)$ represents the space of k-times continuously differentiable functions on Ω [14]:

$$C^k(\Omega) = \{u : D^\alpha u \in C(\Omega) \text{ for all } |\alpha| \leq k\}. \tag{2.1}$$

For $1 \leq p < \infty$, $L^p(\Omega)$ is the space of measurable functions with finite norm defined by

$$\|u\|_{L^p} = \left\{ \int_\Omega |u(x)|^p dx \right\}^{1/p}. \tag{2.2}$$

For integer $m \geq 0$ and $1 < p < \infty$, the Sobolev space

$$W^{m,p}(\Omega) = \{u | D^\alpha u \in L^p(\Omega) \quad \text{for } 0 \leq |\alpha| \leq m\} \tag{2.3}$$

of order (m, p) is the linear space of functions in $L^p(\Omega)$ whose distributional derivatives $D^\alpha u$ of all orders $|\alpha|$ such that $0 \leq |\alpha| \leq m$ are in $L^p(\Omega)$ [14]. For $p = 2$, the notation $H^k(\Omega)$ is used instead of $W^{2,k}(\Omega)$, thus

$$H^k(\Omega) = \{u : D^\alpha u \in L^2(\Omega) \text{ for } |\alpha| \leq k\}. \tag{2.4}$$

In addition to the L^p norms defined in (2.2), the *energy norm* will be used throughout the remaining chapters. The *induced energy norm* is denoted by $|||u||| = \|u\|_a = \sqrt{a(u,u)}$, for all $u \in H_0^1(\Omega)$. $\|u_j\|_V := |A_j^{\frac{1}{2}} u_j|_0$ is the *discrete energy norm* [14]. Finally, the semi-norm on the Sobolev space $W^{m,p}(\Omega)$ is defined as

$$|u|_{m,p,\Omega} = \left[\int_\Omega \sum_{|a|=m} |D^\alpha u(x)|^p dx \right]^{1/p}. \tag{2.5}$$

An *inner product space* $(V, (\cdot,\cdot))$ is a linear space V together with an inner product defined on it. An inner product space is called a *Hilbert space* if the normed linear space $(V, \|\cdot\|)$ is complete. [7]

For a Hilbert space V, V' represents the *dual space* of V. The dual space is the space of continuous linear functionals on V. A *linear functional* on a linear space V is a linear function from V into $\mathbb{R}$; i.e., a function $L : V \to \mathbb{R}$ such that

$L(u + cv) = L(u) + cL(v)$ for all $u, v \in V$, and $c \in \mathbb{R}$. [7]

A bilinear function is a function of two variables which is linear in both variables. A bilinear form $a(\cdot, \cdot)$ on a normed linear space H is said to be *bounded* (or *continuous*) [5] if there exists $0 \leq c < \infty$ such that $|a(v, w)| \leq c\|v\|_H\|w\|_H$ for all $v, w \in H$ and *coercive* or *elliptic* [5] on $V \subset H$ if there exists $\alpha > 0$ such that $a(v, v) \geq \alpha\|v\|_H^2$ for all $v \in V$. A symmetric bilinear form is a bilinear function on a vector space which satisfies $a(u, v) = a(v, u)$.

A function $f : D(\subset \mathbb{R}^n) \to \mathbb{R}^m$ is called *Lipschitz continuous* [5] if there exists $c \in \mathbb{R}$ such that

$$\|f(x) - f(y)\| \leq c\|x - y\| \tag{2.6}$$

for all $x, y \in D$. A *graph* [5] is a hypersurface in $\mathbb{R}^n$ that can be represented by $x_k = f(x_1, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n)$, for $1 \leq k \leq n$. Many boundary value problems are defined on a *Lipschitz domain* [5] $\Omega \subset \mathbb{R}^n$, which is a domain with the property that for every $x \in \partial\Omega$, there is a neighborhood of $\partial\Omega$ which can be represented as the graph of a Lipschitz continuous function.

The terms *a priori* and *a posteriori* will be used in the algorithms described in Chapter 3. An *a priori* decision or strategy is one that is formed beforehand. An *a posteriori* decision, on the other hand, is made based on dynamic observations.

2.2 Boundary value problems

A linear partial differential equation of second order in n variables $x = (x_1, ..., x_n)$ has the form

$$-\sum_{i,k=1}^{n} a_{ik}(x)u_{x_i x_k} + \sum_{i=1}^{n} b_i(x)u_{x_i} + c(x)u = b(x).[13] \tag{2.7}$$

If the functions a_{ik}, b_i and c do not depend on x, then the equation is said to have constant coefficients. For any twice continuously differentiable function $u \in C^2(\Omega)$, $u_{x_i x_k} = u_{x_k x_i}$; thus we can assume that the matrix $A(x) = (a_{ik}(x))$ is symmetric.

Equation (2.7) is called *elliptic* [13] if A is positive definite. A *positive definite* [13] matrix is a symmetric matrix with positive eigenvalues, that is, $v^T Av \geq 0$ for all v. An example of an elliptic partial differential equation is Laplace's equation:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0. \tag{2.8}$$

An example of a *hyperbolic* [13] partial differential equation is the second order wave equation:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}. \tag{2.9}$$

A partial differential equation is hyperbolic if A has one negative and $n-1$ positive eigenvalues.

The third class of partial differential equations are the *parabolic* [13] equations. Equation (2.7) is parabolic if A is positive semi-definite, but not positive definite (i.e., the eigenvalues of A are nonnegative), and the rank (dimension of the range) of

$(A(x), b(x))$ equals n. The heat equation

$$\frac{\partial u}{\partial t} = k\frac{\partial^2 u}{\partial x^2} \tag{2.10}$$

is an example of a parabolic equation.

A partial differential equation, on a given domain, coupled with a set of conditions on the boundary of the domain is called a *boundary value problem* [13]. There are several types of *boundary conditions* [13]. A boundary condition that appears in the boundary value problem explicitly (e.g., $u(0) = 0$) is called *essential* or *Dirichlet* [13]. *Natural* or *Neumann* [13] boundary conditions are conditions on the derivatives on the boundaries and are incorporated implicitly in the boundary value problem. For example, $u'(0) = 0$ is a Neumann boundary condition. The *Robin* [13] boundary condition is of the form

$$\frac{du}{dx} = \pm hu. \tag{2.11}$$

A *periodic* [13] boundary condition is used when a physical problem is defined for $-L < x < L$, but can be thought of as being defined periodically for all x; that is, $x = \hat{x}$ is the same physical point as $x = \hat{x} + 2L$. The periodic boundary conditions can be expressed as

$$\begin{aligned} u(-L) &= u(L), \\ \tfrac{du}{dx}(-L) &= \tfrac{du}{dx}(L). \end{aligned} \tag{2.12}$$

2.3 Discretization methods

2.3.1 Finite differences

The finite difference method is a discretization method used to approximate the solution to an elliptic partial differential equation (PDE) by computing values at points on a rectangular grid covering the domain. The derivatives in the PDE are replaced by divided differences and a linear system of equations is constructed. The divided differences are derived from the Taylor polynomial and the remainder term of the Taylor polynomial determines the *truncation* or *discretization error.* [13].

To illustrate, suppose the PDE is Poisson's equation with Dirichlet boundary conditions in two dimensions:

$$\begin{cases} -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = b(x,y) & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega. \end{cases} \tag{2.13}$$

The second order discretization computes the solution at each point in the grid using the four closest points:

$$4u_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1} = h^2 b_{i,j}. \tag{2.14}$$

Equation (2.14) is called the *five point formula* [14]. This formula can be represented by the *standard five point stencil* [14]:

$$\underbrace{\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}}_{\text{lhs stencil}} \quad \text{and} \quad h^2 \underbrace{\begin{bmatrix} & & \\ & 1 & \\ & & \end{bmatrix}}_{\text{rhs stencil}} \tag{2.15}$$

which indicates the contribution of the neighboring grid points. Here "lhs" represents the left-hand side of equation (2.14) and "rhs" represents the right-hand side of the equation. The following nine point formula results in a fourth order discretization:

$$\begin{aligned} & 20u_{i,j} - 4(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}) \\ & \quad -(u_{i-1,j-1} + u_{i-1,j+1} + u_{i+1,j-1} + u_{i+1,j+1}) \\ & \quad = \tfrac{h^2}{2}(8b_{i,j} + b_{i-1,j} + b_{i+1,j} + b_{i,j-1} + b_{i,j+1}). \end{aligned} \tag{2.16}$$

Equation (2.16) can also be represented in stencil form:

$$\underbrace{\begin{bmatrix} -1 & -4 & -1 \\ -4 & 20 & -4 \\ -1 & -4 & -1 \end{bmatrix}}_{\text{lhs stencil}} \quad \text{and} \quad \tfrac{h^2}{2} \underbrace{\begin{bmatrix} & 1 & \\ 1 & 8 & 1 \\ & 1 & \end{bmatrix}}_{\text{rhs stencil}} \tag{2.17}$$

The truncation error is the difference between the actual solution and the approximation to the solution computed from the discretized system. If the truncation error of the discretization goes to zero as the grid size goes to zero, the approximation is called *consistent* [13].

The linear system of equations resulting from a finite difference discretization can be written in matrix-vector form as $Au = b$. The matrix A is known as the *stiffness* or *system* matrix [5]. This name comes from the corresponding matrices in the context of structural problems. Define A_h as the stiffness matrix on the grid with mesh size h. The approximation satisfies *stability* [13] if the inverses A_h^{-1} remain bounded as h approaches zero. Convergence is guaranteed if consistency and stability are satisfied [13].

The system of equations can be solved by a direct or iterative method. These methods are described in Section 2.4. The matrix A will be sparse; thus solvers that take advantage of the sparsity should be employed. A *sparse matrix* [8] is a matrix with a small number of nonzero entries per row. A *dense matrix* [8], on the other hand, has few zero entries.

2.3.2 Finite elements

The finite element method is based on the variational formulation of an elliptic PDE. Given a boundary value problem, the *variational* or *weak formulation* [7] is constructed as follows: Suppose

1. $(H, (\cdot,\cdot))$ is a Hilbert space,
2. V is a closed subspace of H, and
3. $a(\cdot,\cdot)$ is a bounded, symmetric bilinear form that is coercive on V

Then the *variational problem* is:

$$\text{Given } b \in V', \text{ find } u \in V \text{ such that } a(u,v) = b(v) \text{ for all } v \in V. \tag{2.18}$$

Galerkin's method approximates the solution to the variational problem by defining a discrete problem over the finite dimensional space $V_h \subset V$:

$$\text{Given } b \in V', \text{ find } u_h \in V_h \text{ such that } a(u_h,v) = b(v) \text{ for all } v \in V_h. \tag{2.19}$$

Often the classical solution to an elliptic PDE does not exist, so the generalized solution to the variational problem must suffice. The finite element method is more flexible than the finite difference method and thus can be applied to more complicated problems.

The variational problems have solutions in Sobolev spaces and the solutions are found by minimization in certain finite dimensional linear subspaces of the Sobolev spaces, called finite element subspaces.

To apply the finite element method, the domain Ω is first partitioned into subdomains called *elements* [7], which can be, for example, triangles or quadrilaterals in two dimensions, or tetrahedra or bricks in three dimensions. In this section, the discussion will be restricted to the two dimensional case.

A partition $\mathcal{T} = \{T_1, T_2, ..., T_m\}$ of Ω into triangles or quadrilaterals is called *admissible* [5] if

1. $\displaystyle\bigcup_{i=1}^{m} T_i = \bar{\Omega}$, and

2. If $T_i \cap T_j \neq \emptyset$, then the intersection is either a common vertex or a common edge (but not both) of T_i and T_j. In other words, no vertex of any triangle or quadrilateral lies in the interior of an edge of another triangle or quadrilateral, respectively.

The partition $\mathcal{T}$ is *shape regular* [5] if there exists $\kappa > 0$ such that every triangle (quadrilateral) $T \in \mathcal{T}$ contains a circle of radius $\rho_T \geq h_T/\kappa$, where h_T is half of the diameter of T. The partition $\mathcal{T}$ is *uniform* [5] if there exists $\kappa > 0$ such that every $T \in \mathcal{T}$ contains a circle with radius $\rho_T \geq h/\kappa$, where $2h = \max_{T \in \mathcal{T}} \operatorname{diam} T$. Finally, the partition is *regular* if all the elements are congruent.

A *conforming* [5] triangulation is one in which, for all triangles T_1, T_2 in the triangulation, one of the following holds:

1. $T_1 \cap T_2 = \emptyset$,

2. $T_1 = T_2$, or

3. $T_1 \cap T_2$ is a common vertex or side of T_1 and T_2.

A *conforming finite element* [5] has functions in the Sobolev space in which the variational problem was posed. A triangle (quadrilateral) T contains a *nonconforming point* [5] $p \in \partial T$ if p is a vertex of a neighboring triangle (quadrilateral), but not of T. Such a triangle (quadrilateral) T is called *nonconforming* [5].

On every finite element, piecewise polynomials known as *shape functions* [7] are defined so that each function is associated with a single node of the element. The functions take on the value of 1 on the corresponding node and vanish at all other nodes. Let $\mathcal{P}$ denote the finite dimensional space of shape functions. (Note that the notation $\mathcal{P}_i$ is used to represent the set of all shape functions of degree less than or equal to i.) Then the finite element approximation $v \in \mathcal{P}$ to the solution u of the variational problem (2.18) is given by

$$a(v, \phi) = b(\phi) \;\; \forall \phi \in \mathcal{P}. \tag{2.20}$$

A set $\mathcal{N} = \{\psi_1, \psi_2, ..., \psi_k\}$ of *nodal variables* [7] is constructed as a basis for $\mathcal{P}'$. Using the basis functions, an $n \times n$ (n is the dimension of $\mathcal{P}$) system of linear equations is set up to solve for v. The system of equations is then solved using a direct or iterative method (see Section 2.4).

2.4 Solvers

A linear system of equations $Au = b$ can be solved by direct methods or iterative methods. A *direct method* [8] solves the system exactly in a finite number of steps. Two direct methods often employed are the *Gaussian elimination* and *Cholesky factorization* techniques [8].

To solve the system using Gaussian elimination, A is factored into the product of a lower and an upper triangular matrix: $A = LU$, where L is lower triangular (typically with 1's along the diagonal) and U is upper triangular. Thus the system $Au = b$ becomes $LUu = b$. Setting $Uu = y$, the system $Ly = b$ is first solved for y using forward substitution, then $Uu = y$ is solved for u using backward substitution.

A may also be factored into $A = LDU$, where D is a diagonal matrix. Solving the system $LDUu = b$ is similar to solving $LUu = b$.

If A is symmetric positive definite, then the Cholesky factorization can be used: $A = U^T U$, where U is upper triangular. Forward and backward substitution are again used to solve the system $U^T U u = b$.

One drawback of the direct solvers is that the triangular matrices are often dense independent of the sparsity of the original matrix A. The *Incomplete LU Decomposition* (ILU) [14] can be used to eliminate calculation of the matrix entries in the factorization which are zero in the original matrix A. The ILU method factors A into $A = LU - C$, where L and U are upper and lower triangular matrices, respectively, which have the same sparsity pattern as A.

Another problem arising from the application of direct methods is the error produced when digits are rounded off.

Iterative methods [8] solve the system of equations by generating a sequence of iterates u^k, starting with an initial guess u^0, which converges to the solution as k approaches infinity. To set up the iteration, the matrix A is decomposed into $A = M - N$, where M is an easily invertible matrix. The system $Au = b$ can then be written as $Mu = Nu + b$, which gives us the iteration $Mu^{k+1} = Nu^k + b$. Thus

$$u^{k+1} = M^{-1}(Nu^k + b) \tag{2.21}$$

where the solution u of the original system is a fixed point of equation (2.21); that is, $u = M^{-1}(Nu + b)$. The Jacobi and Gauss-Seidel iterative methods are among the classical iterative methods.

The matrix A can be expressed as $A = D - L - U$ (i.e., $M = D$ and $N = L + U$), where D is the diagonal of A, and L and U are strictly lower and upper triangular matrices, respectively. The components of this decomposition are used in the Jacobi (2.23) and Gauss-Seidel (2.31) iterations.

The *Jacobi iteration* [14] updates the approximation sequentially; that is, the components of the vector u^{k+1} are determined using the components of u_k, as follows:

$$a_{ii} u_i^{k+1} = -\sum_{j \neq i} a_{ij} u_j^k + b_i, \tag{2.22}$$

for $i = 1, ..., n$.

In matrix form, the Jacobi iteration is represented by

$$D u^{k+1} = (L + U) u^k + b. \tag{2.23}$$

Thus the updated vector can be written as

$$u^{k+1} = u^k - D^{-1}(Au^k - b) = D^{-1}[(L + U)u^k + b]. \tag{2.24}$$

The *damped Jacobi iteration* [14], which reduces or damps the change in u^k, can be written as

$$u^{k+1} = u^k - \theta D^{-1}(Au^k - b), \tag{2.25}$$

where $\theta \in (0, 1)$. A common θ is $\lambda_{\max}^{-1}$ where $\lambda_{\max}$ is the largest eigenvalue of A.

As the damped Jacobi iteration is a linear function of u and b, it can be represented as

$$S^{DJ}(u, b) = Mu + Nb, \tag{2.26}$$

where $M = I - \theta D^{-1}A$ and $N = \theta D^{-1}$.

The Jacobi and damped Jacobi iterations are independent of the ordering of the unknowns. The diagonal entries of A are typically of order $\mathcal{O}(h^{2m})$, where $2m$ is the order of $Au = b$. Thus if the diagonal elements are constant, $\theta D^{-1} = \omega h^{2m}$. Equation (2.25) becomes

$$u^{k+1} = u^k - \omega h^{2m}(Au^k - b), \tag{2.27}$$

with $M = I - \omega h^{2m}A$ and $N = \omega h^{2m}I$ in equation (2.26), where I is the identity matrix. One possible choice for ω is $\omega = \|h^{2m}A\|^{-1}$, where $\|\cdot\|$ is the spectral norm.

The ω in the damped Jacobi iteration given in (2.27) is constant. If the ω is allowed to vary, the method is known as Richardson's iteration.

The Jacobi iteration updates the approximation sequentially, using only the values of the vector u^k to determine u^{k+1}. The *Gauss-Seidel iteration*, or *method of successive relaxation* [14], uses those components of u^{k+1} already computed to update the remaining components of the vector. For example, given a natural ordering of the nodes, when the iteration arrives at the computation of u_i^{k+1}, the components u_j^{k+1} for $j = 1, ..., i-1$ have already been computed and can be used in the current computation:

$$a_{ii}u_i^{k+1} = -\sum_{j<i} a_{ij}u_j^{k+1} - \sum_{j>i} a_{ij}u_j^k + b_i. \tag{2.28}$$

The components of u^k can be written over by the components of u^{k+1} as they are computed; thus the Gauss-Seidel method requires no extra storage space, unlike the Jacobi method, which requires an extra vector for u^{k+1}.

In matrix form, the Gauss-Seidel iteration is

$$Du^{k+1} = Lu^{k+1} + Uu^k + b; \tag{2.29}$$

thus

$$u^{k+1} = (D - L)^{-1}[Uu^k + b]. \tag{2.30}$$

The Gauss-Seidel iteration converges twice as fast as the Jacobi iteration [24]. A variation of the Gauss-Seidel method involves two iterations. After the components of the vector u^{k+1} have been updated, a backward sweep of the Gauss-Seidel iteration can be performed, thereby updating the components in reverse order. The coupling of the forward and backward sweeps is known as the *symmetric Gauss-Seidel iteration* (SGS) [14]. One symmetric Gauss-Seidel iteration costs twice that of the standard Gauss-Seidel iteration. The convergence of SGS is slightly better than that of the standard iteration and yields the added benefit of a symmetric operator.

The symmetric Gauss-Seidel iteration can be expressed as

$$\begin{aligned} &\text{For } i = 1, 2, \ldots, n-1, n, n-1, \ldots, 1 \\ &\quad u_{j,i} = \left[\sum_{k=1, k\neq i}^{n} A_{j,ik} u_{j,k} - b_{j,i} \right] / A_{j,ii} \end{aligned} \tag{2.31}$$

In matrix form, (2.31) can be expressed as

$$
\begin{aligned}
\hat{u}^{k+1} &= (D-L)^{-1}(Uu^k + b) \\
u^{k+1} &= (D-U)^{-1}(L\hat{u}^{k+1} + b).
\end{aligned}
\tag{2.32}
$$

Thus the symmetric Gauss-Seidel iteration is

$$
S^{SGS}(u, b) = Mu + Nb, \tag{2.33}
$$

where the iteration matrix is

$$
\begin{aligned}
M &= (D-U)^{-1}L(D-L)^{-1}U \\
&= I - [(D-L)D^{-1}(D-U)]^{-1}A,
\end{aligned}
\tag{2.34}
$$

and

$$
N = (D-U)^{-1}[L(D-L)^{-1} + I]. \tag{2.35}
$$

The k^{th} iterate u^k can be expressed as

$$
u^k = S_k^{SGS}u^0 = M^k u^0 + N^{(k)}b, \tag{2.36}
$$

where $N^{(k)} = \displaystyle\sum_{i=0}^{k-1} M^i N$. The solution u is a fixed point of (2.36), thus $u = Mu + Nb$, or $I = M + NA$. If A is non-singular, then the matrix N can be written as $N = (I-M)A^{-1}$.

The Symmetric Gauss-Seidel iteration can also be written as

$$W(u^k - u^{k+1}) = Au^k - b, \tag{2.37}$$

where $W = (D - L)D^{-1}(D - U)$. If $A = A^* > 0$ and $D = D^* > 0$, then W is symmetric positive definite and satisfies $0 < A \leq W$.

One drawback of the Jacobi and Gauss-Seidel methods is the poor reduction of the low frequency error components. Accelerated versions of the Gauss-Seidel iteration are the successive over relaxation (SOR) and symmetric successive over relaxation (SSOR) methods. These accelerated methods work better on the low frequency components, but do not work as well on the higher frequencies.

To derive the SOR and SSOR methods, the Gauss-Seidel iteration (2.28) can be written as

$$a_{ii}u_i^{k+1} = \omega\left[-\sum_{j<i} a_{ij}u_j^{k+1} - \sum_{j>i} a_{ij}u_j^k + b_i\right] - (\omega - 1)a_{ii}u_i^k, \tag{2.38}$$

with $\omega = 1$. The value $\omega \in (0, 2)$ is known as the *relaxation parameter*. Changing the value of ω can make the iteration converge faster. If $\omega > 1$, the iteration (2.38) is called the *successive overrelaxation* (SOR) [14] iteration. For $\omega < 1$ we have an *underrelaxation* (SUR) [14], however, the iteration is referred to as overrelaxation for all $\omega \in (0, 2)$. Equation (2.38) in matrix form can be expressed as

$$Du^{k+1} = \omega\left[Lu^{k+1} + Uu^k + b\right] - (\omega - 1)Du^k. \tag{2.39}$$

In the *symmetric successive overrelaxation* (SSOR) [14] method, the iteration is performed in the forward and backward directions, just like the symmetric Gauss-Seidel iteration. Equation (2.40) illustrates the SSOR iteration in matrix form:

$$
\begin{aligned}
D\hat{u}^{k+1} &= \omega\left[L\hat{u}^{k+1} + Uu^{k} + b\right] - (\omega - 1)Du^{k}, \\
Du^{k+1} &= \omega\left[L\hat{u}^{k+1} + Uu^{k+1} + b\right] - (\omega - 1)D\hat{u}^{k+1}.
\end{aligned}
\tag{2.40}
$$

The computational work for k cycles of the SSOR method is equal to the work for $k + \frac{1}{2}$ SOR cycles, rather than $2k$ [5]. The additional work is insignificant in comparison to the added benefit of symmetry of the iteration matrix in the SSOR method, which allows it to be an effective preconditioner for other iterative techniques on symmetric matrices.

The conjugate gradient method is a semi-iterative method [16]. Suppose $L = L^* > 0$ and $W = W^* > 0$. The preconditioned conjugate gradient method is defined in Algorithm 2.1. If $W = I$, then Algorithm 2.1 is the original conjugate gradient method [14].

The matrix W is known as a *preconditioner* [5]. A preconditioner is a matrix W which approximates the matrix A, is positive definite, and is easy to invert. If u_0 is an initial approximation to the solution of the system $Au = b$, then an updated approximation u_1 can be computed as follows: $u_1 = u_0 - cW^{-1}g_0$, where $g_0 = Au_0 - b$. If $W = A$ and $c = 1$, then u_1 is the exact solution. Using a good approximation to A as the preconditioner allows a more rapid convergence to the solution u.

Algorithm 2.1 Preconditioned conjugate gradient method

1. Let u^0 be the starting value
2. $r^0 = b - Au^0$
3. $p^0 = W^{-1}r^0$
4. For $k = 0, ..., m$
   (a) $q^k = Ap^k$
   (b) $\alpha_k = \frac{(r^k, p^k)}{(p^k, q^k)}$
   (c) $u^{k+1} = u^k + \alpha_k p^k$
   (d) $r^{k+1} = r^k - \alpha_k q^k$
   (e) $v^{k+1} = W^{-1}r^{k+1}$
   (f) $\beta_k = \frac{-(q^k, v^{k+1})}{(p^k, q^k)}$
   (g) $p^{k+1} = v^{k+1} + \beta_k p^k$

Standard preconditioners are based on the diagonal, SGS, Jacobi, SSOR, and variants of ILU. Many times, an iterative procedure is called to do one iteration of the preconditioner.

2.5 Multigrid methods

Multigrid was originally developed to solve boundary value problems arising in physical applications. A set of grid points is chosen in the domain of the problem. Using a discretization process, a system of algebraic equations arises that is associated with the chosen grid points and whose solution is an approximation to the continuous problem. [14]

A sequence of related problems is generated on coarser and coarser grids, where the solution on a particular grid is a good approximation to the solution on the next finer grid. Alternately, a coarse grid problem is solved and then (adaptively or uniformly) refined until a fine enough grid is generated that adequately solves the problem.

A restriction operator maps an approximate solution on one level to the next coarser level. This can be, for example, a direct injection at coincident points or a weighted average of each grid point with its nearest neighbors. An interpolation or prolongation operator takes an approximate solution and maps it to the next finer grid. For example, on a triangular partition, the interpolation operator I_j maps values in the finite dimensional vector space X_j to the space X_{j+1} as follows: Let a and b be two neighboring nodes in the j^{th} triangulation (see Figure 2.1).

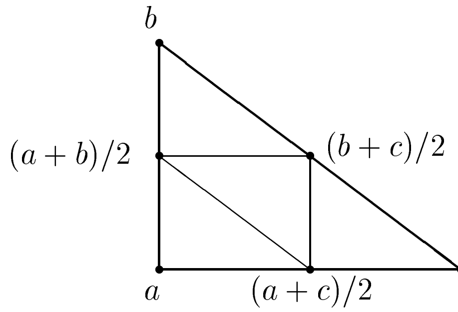

Figure 2.1: j^{th} and $(j+1)^{st}$ triangulations

Then $u_{j+1}(a) = u_j(a)$ and $u_{j+1}(b) = u_j(b)$. If $(a+b)/2$ is a node in the $(j+1)^{st}$ triangulation, then $u_{j+1}((a+b)/2) = (u_j(a) + u_j(b))/2$; that is, the value at the new node is the average of the neighboring nodes.

The V cycle is the basic multigrid algorithm.

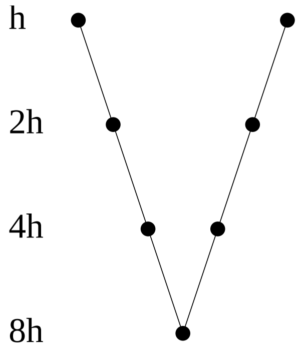

Figure 2.2: Four level multigrid V cycle

The problem starts on the finest level. A smoother (e.g., Gauss-Seidel) or rougher (e.g., conjugate gradients) is used on this level to damp the high frequency components of the error. [14] The residual is computed and projected to the next coarser level where the residual problem is solved with an initial guess of zero.

This process is repeated down to the coarsest level, where the solution to the residual problem is computed (typically with a direct solver). Then the solution is corrected on the next finer grid by adding an interpolation of the coarse grid solution to the existing solution on this finer level. On the next finer level, the problem is smoothed and the new correction is interpolated up to the next finer level where again the existing solution is corrected. This continues until the finest level is reached, where the approximation is smoothed again to get the final solution.

The full multigrid cycle (FMG) is composed of a series of nested V cycles [14].

Algorithm FMG begins on the coarsest level. The idea here is to get a good initial guess for a particular level before doing any processing on that level. The cascadic

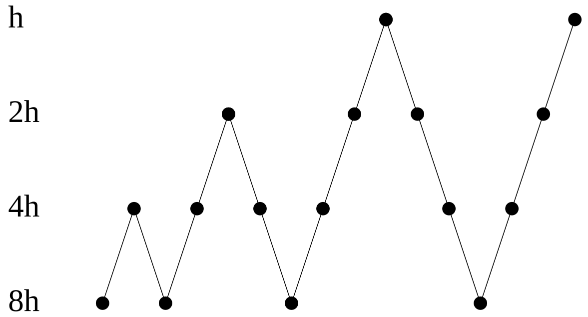

Figure 2.3: Four level full multigrid cycle

multigrid algorithm differs from the full multigrid cycle in that it is a one way multigrid starting at the coarsest level and continues to the finest level without ever going back for a correction.

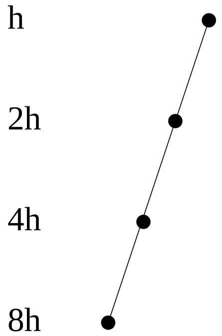

Figure 2.4: Four level Cascadic multigrid cycle

A prime feature of cascadic multigrid is that more smoothing iterations are done on the coarser levels in the hope that fewer smoothing iterations are needed on the finer levels; i.e., we *over solve* the problem on the coarser levels.

Once the continuous problem is discretized on all of the levels, the smoother is run on each level only until the residual norm is reduced to some (fixed) size (e.g.,

10^{-4} or 10^{-6}). The problem is over solved in the hope that the initial guess is better when it is interpolated to the finer levels.

Chapter 3 discusses various cascadic multigrid algorithms.

Chapter 3

Cascadic Algorithms

In this chapter the evolution of the cascadic multigrid method is described. The algorithms described in sections 3.1 through 3.6 apply to elliptic problems and the algorithms in section 3.7 are used to solve parabolic problems. In addition to the algorithms are some of the applications of the cascadic multigrid technique.

3.1 The Cascadic Principle

Cascadic multigrid evolved from the *Cascadic Principle* of Deuflhard, Leinen, and Yserentant [11]. The cascadic multigrid algorithm differs from the full multigrid cycle in that it is a one way multigrid starting at the coarsest level and continuing to the finest level without ever going back to a previous level for a correction.

Deuflhard, Leinen, and Yserentant developed an adaptive finite element code called KASKADE to solve elliptic boundary problems on general domains using hierarchical bases. This approach achieves the same level of computational complexity as the standard multigrid method, but can be applied to non-uniform grids.

The algorithm which follows illustrates the main features of the KASKADE code. In this algorithm, Cholesky decomposition (or another sparse elimination technique) is used to solve the problem on the coarsest grid (level 0). The solution u_0 is in the space of piecewise linear finite elements of dimension n_0 associated with the initial triangulation. To get an approximation on the next finer level, the triangulation of the previous level is first refined until the number of vertices has at least doubled. This refinement is based on an edge oriented local error estimator which determines the edges to be refined.

The hierarchical basis of the finite element space is then constructed, as follows [25]. Let $\mathcal{V}_1$ represent the set of nodes (vertices) $x_1, ..., x_{n_1}$ of the partition $\mathcal{T}_1$ on level 1 and let $\mathcal{P}^{(1)}$ represent the finite element space corresponding to $\mathcal{T}_1$. $\mathcal{P}^{(1)}$ is spanned by the nodal basis functions $\psi_i^{(1)} \in \mathcal{P}^{(1)}, i = 1, ..., n_1$ defined by

$$\psi_i^{(1)}(x_k) = \delta_{ik}, \quad x_k \in \mathcal{V}_1. \tag{3.1}$$

Then the *hierarchical basis functions*[25] are defined as

$$\begin{aligned} \hat{\psi}_i &= \psi_i^{(0)}, \qquad x_i \in \mathcal{V}_0, \\ \hat{\psi}_i &= \psi_i^{(1)}, \qquad x_i \in \mathcal{V}_1 \backslash \mathcal{V}_0. \end{aligned} \tag{3.2}$$

In the hierarchical basis multigrid method, only the points in $\mathcal{V}_1 \backslash \mathcal{V}_0$ are smoothed on level 1; thus the operation count remains $\mathcal{O}(n)$.

To get the initial approximation in the piecewise linear finite element space of dimension n_1, all hierarchical basis coefficients associated with u_0 are maintained and the new coefficients are set to zero. Finally, the preconditioned conjugate gradient method is used on the approximation until the algebraic error is below a predetermined constant multiple of the discretization error. The *algebraic error* is given by $\|u_j - u_j^*\|$ and the *discretization error* is $\|u - u_j\|$, where u is the solution, u_j is the finite element approximation to u on level j, and u_j^* is the iterative approximation.

The process on level 1 is repeated for all finer levels, where each triangulation and initial approximation is based on that of the level immediately preceding the current level.

Algorithm 3.1 Cascadic Principle

---

1. If level $j = 0$
   (a) Determine initial triangulation
   (b) Use a sparse elimination technique to determine the solution u_0
2. If level $j > 0$
   (a) Refine the previous triangulation
   (b) Construct the hierarchical basis of the finite element space associated with the triangulation
   (c) Determine the initial approximation in the piecewise linear finite element space of dimension n_j
   (d) Smooth using the preconditioned conjugate gradient method

---

All computations in this algorithm are local; thus it is easily parallelized.

The KASKADE code was used to solve problems involving re-entrant corners, discontinuous Dirichlet boundary conditions, and discontinuous coefficients.

3.2 Cascadic Conjugate Gradient Method

3.2.1 Cascadic Preconditioned Conjugate Gradient Method

The cascadic preconditioned conjugate gradient (CPCG) method for elliptic boundary value problems is derived from the cascadic principle [10]. The revision simplifies the original method and makes it more robust. In the CPCG method, the linear system on the coarsest grid is solved directly by sparse elimination techniques, while the problem on the finer grids is solved using preconditioned conjugate gradient iterations. The starting values on the finer levels are the approximate finite element solutions from the previous levels. The finite element spaces on each level are constructed adaptively, based on the energy norm error estimators.

In this approach, Galerkin finite element methods are used for the outer iterations and preconditioned conjugate gradient methods are used for the inner iterations. Both methods minimize the energy norm of the iteration errors. The Galerkin methods minimize the error over the finite element spaces while the conjugate gradient methods minimize over the Krylov spaces. (A Krylov space [19] $\mathcal{K}_m$ is defined to be the m-dimensional subspace of $\mathbb{R}^n$ defined by

$$\mathcal{K}_m = \text{span}(r_0, Ar_0, A^2r_0, ..., A^{m-1}r_0), \tag{3.3}$$

where $r_0 = b - Au_0$ for initial guess u_0. A solution to $Au = b$ is sought in the subspace $u_0 + \mathcal{K}_m$.)

Algorithm 3.2 shows the steps of the preconditioned conjugate gradient method applied to a linear system $Au = b$, with A symmetric positive definite and $B \sim A^{-1}$ a symmetric positive definite preconditioner. Here u_0 is the initial approximation and $u_{j,k}$ are the preconditioned conjugate gradient iterates. The index j represents the level on which the iteration is performed. In the conjugate gradient algorithm, as well as in all cascadic algorithms to follow, more iterations are performed on the coarser levels so that the number of iterations on the finer levels will be smaller.

Algorithm 3.2 Cascadic preconditioned conjugate gradient method

1. If $j = 0$
   (a) Solve directly for u_0
2. If $j > 0$
   (a) Set $u_{j,0} = u_{j-1}$
   (b) For $k = 1, ..., n_j$
      i. Smooth using preconditioned conjugate gradient on $u_{j,k-1}$ to get $u_{j,k}$ (inner iteration)
   (c) Smooth using Galerkin finite element methods (outer iteration) to obtain u_j

$A_{j-1} = P^T A_j P$ is the Galerkin formulation for the coarse grid problem, where P is the prolongation.

Numerical experiments were performed using two cascadic preconditioned conjugate gradient modes: the adaptive mode, which includes an adaptive mesh refinement strategy, and the uniform mode. Within each mode, three types of preconditioners were examined: hierarchical bases, multilevel preconditioners, and no preconditioner.

In the adaptive mode, for both a peak problem and a re-entrant corner problem, it was determined that the case without the preconditioner was the simplest and fastest. In the uniform mode, the case without the preconditioner performed the best again. This appears to be a result of some preconditioning inherent in the cascadic method.

3.2.2 Cascadic Conjugate Gradient Method Without Preconditioning

The cascadic conjugate gradient method (CCGM) is the CPCG method without preconditioning [21]. It is accurate with optimal complexity for elliptic problems with H^2-regularity in two dimensions with quasi-uniform triangulations.

In Algorithm 3.3, a direct solve is performed on the coarsest level and the conjugate gradient method without preconditioning is performed on all subsequent levels.

The number of iterations on each level is predetermined based on the level of accuracy desired in the operation. Recall that the number of iterations of the conjugate gradient method decreases as the grids become finer.

Algorithm 3.3 Cascadic conjugate gradient

---

1. If $j = 0$
   - (a) Perform a direct solve to find u_0: $u_0 = A_0^{-1} b_0$
2. If $j > 0$
   - (a) Interpolate to get w_j: $w_j = P_{j-1} u_{j-1}$
   - (b) $y_0 = w_j$
   - (c) Perform m_j iterations of the conjugate gradient algorithm on y_0 using Algorithm 2.1 with $W = I$ to get y_{m_j}
   - (d) $u_j = y_{m_j}$

Stokes problem

The Stokes problem represents the motion of an incompressible viscous fluid in a two or three dimensional domain. The problem is to find $u \in [C^2(\Omega) \cap C^0(\bar{\Omega})]^n$ for $n = 2, 3$ and $p \in C^1(\Omega)$ such that

$$\begin{aligned} \Delta u + \text{grad } p &= -b \quad \text{in } \Omega, \\ \text{div } u &= 0 \quad \text{in } \Omega, \\ u &= u_0 \quad \text{on } \partial\Omega, \end{aligned} \tag{3.4}$$

where u is the velocity field and p is the pressure.

Define $X = H_0^1(\Omega)^d$ for $d = 2, 3$ and $M = L_{2,0}(\Omega) = \{q \in L_2(\Omega) : \int_\Omega q dx = 0\}$. The variational form of the Stokes problem (3.4) can be expressed as: Find $u \in X$ and $p \in M$ such that

$$\begin{aligned} a(u, v) + f(v, p) &= (b, v) \quad \forall\, v \in X, \\ f(u, q) &= 0 \qquad \forall\, q \in M, \end{aligned} \tag{3.5}$$

where

$$\begin{aligned} a(u, v) &= \int_\Omega \nabla u \nabla v \, dx, \\ f(v, q) &= -\int_\Omega \text{div } v \, q \, dx. \end{aligned} \tag{3.6}$$

The triangulations T_j are assumed to be shape-regular and the finite element spaces are nested:

$$\begin{aligned} &X_0 \subset X_1 \subset \cdots \subset X_l \subset X, \text{ and} \\ &M_0 \subset M_1 \subset \cdots \subset M_l \subset M. \end{aligned} \tag{3.7}$$

Equation (3.5) can be expressed on level j as

$$\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u_j \\ q_j \end{pmatrix} = \begin{pmatrix} b_j \\ 0 \end{pmatrix}, \tag{3.8}$$

where

$$\begin{aligned} (Au_j, v) &= a(u_j, v) \quad \forall\, v \in X_j, \\ (Bu_j, q) &= f(u_j, q) \quad \forall\, q \in M_j. \end{aligned} \tag{3.9}$$

The matrix A is symmetric positive definite. Let W be a preconditioner satisfying

$$v^T A v \leq v^T W v, \tag{3.10}$$

and for which

$$\begin{pmatrix} W & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} v \\ q \end{pmatrix} = \begin{pmatrix} d \\ e \end{pmatrix}, \tag{3.11}$$

is easier to solve than (3.8).

The inverse of $\begin{pmatrix} W & B^T \\ B & 0 \end{pmatrix}$ is

$$\begin{pmatrix} W^{-1}(I - B^T S^{-1} B W^{-1}) & W^{-1} B^T S^{-1} \\ S^{-1} B W^{-1} & -S^{-1} \end{pmatrix} \tag{3.12}$$

where $S = BW^{-1}B^T$ is the Schur complement of (3.12).

For the saddle point problem, $W = \gamma I$, thus $v^T A v \le \gamma v^T v$ and (3.12) becomes

$$\begin{pmatrix} \gamma I & B^T \\ B & 0 \end{pmatrix}^{-1} = \begin{pmatrix} \frac{1}{\gamma} P & B^T(BB^T)^{-1} \\ (BB^T)^{-1}B & -\gamma(BB^T)^{-1} \end{pmatrix}, \tag{3.13}$$

where $P = I - B^T(BB^T)^{-1}B$ is the projection to $V = \ker B$.

Thus the iteration is

$$\begin{pmatrix} u^{k+1} \\ p^{k+1} \end{pmatrix} = \begin{pmatrix} u^k \\ p^k \end{pmatrix} - \begin{pmatrix} \gamma I B^T \\ B \end{pmatrix}^{-1} \left\{ \begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u^k \\ p^k \end{pmatrix} - \begin{pmatrix} b \\ g \end{pmatrix} \right\}, \tag{3.14}$$

where u^{k+1} satisfies $Bu^{k+1} = g$. To perform the iteration, (3.11) must be solved (with $W = \gamma I$). Thus

$$\begin{aligned} BB^T q &= Bd - \gamma e, \\ v &= \tfrac{1}{\gamma}(d - B^T q), \end{aligned} \tag{3.15}$$

which is similar to solving Poisson's equation for the Stokes problem.

Conjugate directions can be constructed from (3.14) as follows: Define the residual vector $g^k = Au^k + B^T p^k - b$ and compute h^k from

$$\begin{pmatrix} \gamma I & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} h^k \\ p^k \end{pmatrix} = \begin{pmatrix} g^k \\ 0 \end{pmatrix}. \tag{3.16}$$

Set $d^k = -h^k + \beta_k d^{k-1}$ and $u^{k+1} = u^k + \alpha_k d^k$, where α_k and β_k are defined as in Algorithm 2.1.

Algorithm 3.4 gives the cascadic multigrid using conjugate gradients for the Stokes problem.

Algorithm 3.4 Cascadic multigrid for the Stokes equations

1. If $j = 0$
   (a) Compute exact solution u_0, q_0
   (b) Set $v_0 = u_0$
2. If $j > 0$
   (a) Interpolate v_{j-1} to get w_j
   (b) Compute $v^0 := v_j^0 = Pw_j$ (the projection of w_j to $V_j = \ker B_j$)
   (c) For $k = 1, ..., m_j$
      i. Smooth using the conjugate gradient iteration
   (d) Set $v_j = v^{m_j}$

Theorem 3.2.1 *[6] For the cascadic multigrid for the Stokes problem with* $1 < \gamma < d$, $d = 2, 3$, *and* $m_j = \lceil m_l 2^{\gamma(l-j)} \rceil$ *conjugate gradient iterations on level* j, $j = 0, \ldots, l$,

$$\|u_l - v_l\|_1 \leq c \frac{h_l}{m_l} \|b\|_0. \tag{3.17}$$

The complexity of the algorithm is bounded by $c\, m_l n_l$, *where* $n_l = \dim X_l$.

3.3 Cascadic Multigrid with General Smoothers

The cascadic multigrid method was extended to include linear energy reducing smoothers such as symmetric Gauss-Seidel, SSOR, and damped Jacobi [3].

Let S_{j,m_j} denote m_j iterations of a smoother such as symmetric Gauss-Seidel, SSOR, or damped Jacobi on level j. Define $\mathfrak{S}_j^{m_j} : V_j \to V_j$ as the error propagation, where

$$u_j - S_{j,m_j} u_j^0 = \mathbf{S}_j^{m_j}(u_j - u_j^0). \tag{3.18}$$

Then an *energy reducing smoother* S_{j,m_j} is one for which the *smoothing properties* (3.19) hold:

$$\begin{aligned} (i) \quad |||\mathbf{S}_j^{m_j} v_j||| &\leq c\frac{h_j^{-1}}{m_j^{\gamma}}\|v_j\|_0, \\ (ii) \quad |||\mathbf{S}_j^{m_j} v_j||| &\leq |||v_j|||, \end{aligned} \tag{3.19}$$

for all v_j in the space V_j, where $0 < \gamma \leq 1$. For the symmetric Gauss-Seidel, SSOR, and damped Jacobi iterations, $\gamma = \frac{1}{2}$.

3.3.1 Poisson's equation with P1 conforming element

The elliptic problem to be solved is Poisson's equation with Dirichlet boundary conditions:

$$\begin{aligned} -\Delta u &= b \quad \text{in } \Omega, \\ u &= 0 \quad \text{on } \partial\Omega, \end{aligned} \tag{3.20}$$

where Δ denotes the Laplace operator

$$\Delta := \sum_{i=1}^{n} \frac{\partial^2}{\partial x_i^2}. \tag{3.21}$$

The weak formulation of the problem is:

$$\text{Find } u \in H_0^1 \text{ such that } a(u,v) = b(v), \quad \forall\, v \in H_0^1. \tag{3.22}$$

Here b is a bounded linear functional on $H^1(\Omega)$ and a is an $H_0^1(\Omega)$-elliptic bilinear form; that is, $a(u,u) \geq c||u||^2_{H_0^1}$ for all $u \in H_0^1$ with $c > 0$.

To approximate u, a nested family of triangulations of Ω, $\mathcal{T}_0 \subset \mathcal{T}_1 \subset \cdots \subset \mathcal{T}_l$, is used. The spaces of linear finite elements are

$$V_j = \{u \in C(\bar{\Omega}) : u|_T \in \mathcal{P}_1(T), \quad \forall\, T \in \mathcal{T}_j, \quad u|_{\partial\Omega} = 0\}, \tag{3.23}$$

where $\mathcal{P}_1(T)$ denotes the linear functions on the triangle T. The triangulations are *quasi-uniform* with mesh size parameter

$$\frac{1}{c} 2^{-j} \leq h_j = \max_{T \in \mathcal{T}} \text{ diam } T \leq c\, 2^{-j}, \tag{3.24}$$

where c is any positive constant, depending only on the bilinear form $a(\cdot,\cdot)$, on Ω and on the shape regularity as well as the quasi-uniformity of the triangulations.

These elements are called *P1 conforming elements*, or *Courant triangles* [5]. A P1 conforming element is an element whose finite element space consists of piecewise linear functions defined at the vertices of T.

The spaces V_j satisfy $V_0 \subset V_1 \subset \cdots \subset V_l \subset H_0^1(\Omega)$. The finite element approximations are thus given by:

$$\text{Find } u_j \in V_j \text{ such that } a(u_j, v_j) = b(v_j), \quad \forall\, v_j \in V_j. \tag{3.25}$$

$H^{1+\alpha}$-regularity is assumed on the elliptic problem:

$$\|u\|_{1+\alpha} = \sum_{\nu \leq |1+\alpha|} \left(\int |D^{\nu} u|^2 \right)^{\frac{1}{2}} \leq c \, \|b\|_{\alpha-1}, \quad \forall \, b \in H^{\alpha-1}(\Omega), \tag{3.26}$$

for some $0 < \alpha \leq 1$. The approximation error of the finite element method is as follows:

$$|||u - u_j||| \leq c h_j^{\alpha} \|b\|_{H^{\alpha-1}}, \quad j = 0, \ldots, l, \tag{3.27}$$

where $||| \cdot |||$ represents the energy norm. Here,

$$\|b\|_{H^{\alpha-1}} = \left(\int_{\Omega} \sum_{|k| \leq \alpha-1} \left(D^k b \right)^2 dx \right)^{\frac{1}{2}} .[9]$$

Céa's Theorem

A useful tool in proving convergence of the finite element approximation u_j to the exact solution u is Céa's Theorem. This theorem allows us to estimate the error $u - u_j$ by estimating the distance of u from the subspace V_j, which is denoted by $\inf\limits_{v_j \in V_j} \|u - v_j\|$. The interpolant $\tilde{u}_h \in V_h$ of u is a useful choice in the estimate.

The following is Céa's Theorem, summarized from [5], [7] and [20].

Theorem 3.3.1 *(Céa's Theorem) Suppose V is a closed subset of a Hilbert space, V_h is a finite-dimensional subspace of V, and a is a continuous V-elliptic bilinear form. Let u and u_h represent the solutions of the variational problem in V and V_h,* *respectively. Then there is a constant c such that*

$$\|u - u_h\|_V \leq c \min_{v \in V_h} \|u - v\|_V,$$

where $\|\cdot\|_V$ *represents the norm on the space* V*.*

Approximation Property

By duality and an interpolation argument one gets the *approximation property* for $u \in H^{1+\alpha}$ and $0 \leq \alpha \leq 1$:

$$\|u_j - u_{j-1}\|_{1-\alpha} \leq c h_j^{\alpha} |||u_j - u_{j-1}|||, \quad j = 1, \ldots, l. \tag{3.28}$$

In the remainder of this section, I show this property by expanding on methods illustrated in [1], [2] and [5]. Let $\lambda_i^{(j)}$ and $\psi_i^{(j)}$, $1 \leq i \leq n_j = \dim V_j$, represent the eigenvalues and eigenvectors, respectively, associated with the space V_j that satisfy

$$a(\psi_i^{(j)}, v_j) = \lambda_i^{(j)} b_j(\psi_i^{(j)}, v_j) \quad \forall\, v_j \in V_j$$

where

$$0 < \lambda_1^{(j)} \leq \lambda_2^{(j)} \leq \cdots \leq \lambda_{n_j}^{(j)},$$

$$b_j(\psi_i^{(j)}, \psi_k^{(j)}) = \delta_{ik}, \text{ and}$$

$$a(\psi_i^{(j)}, \psi_k^{(j)}) = \lambda_i^{(j)} \delta_{ik},$$

where $b_j(\cdot,\cdot)$ is a symmetric positive definite bilinear form comparable to the L^2 inner product, and δ_{ik} is the Kronecker delta (cf. [2] and [1]). Define the discrete norms $|||\cdot|||_s$, $-2 \le s \le 2$, as follows: $|||v|||_s^2 = \sum_{i=1}^{n_j} c_i^2 (\lambda_i^{(j)})^s$ for $v = \sum_{i=1}^{n_j} c_i \psi_i^{(j)}$. Thus $|||v|||_1 = |||v|||$ and $|||v|||_0$ is comparable to $\|v\|_{L^2}$, as shown in the following:

$$
\begin{aligned}
|||v|||^2 &= a(v,v) \\
&= a\left(\sum_i c_i\ \psi_i^{(j)}, \sum_k c_k\ \psi_k^{(j)}\right) \\
&= \sum_i c_i\ a\left(\psi_i^{(j)}, \sum_k c_k\ \psi_k^{(j)}\right) \\
&= \sum_i c_i \sum_k c_k\ a\left(\psi_i^{(j)}, \psi_k^{(j)}\right) \\
&= \sum_i c_i \sum_k c_k\ \lambda_i^{(j)} b_j\left(\psi_i^{(j)}, \psi_k^{(j)}\right) \\
&= \sum_i c_i\ (c_i\ \lambda_i^{(j)}) \\
&= \sum_i c_i^2\ \lambda_i^{(j)} \\
&= |||v|||_1^2.
\end{aligned}
\tag{3.29}
$$

We have $c^{-1}\ (v,v) \le b_j(v,v) \le c\ (v,v)$ for some constant c and

$$
\begin{aligned}
b_j(v,v) &= b_j\left(\sum_i c_i\ \psi_i^{(j)}, \sum_k c_k\ \psi_k^{(j)}\right) \\
&= \sum_i c_i\ b_j\left(\psi_i^{(j)}, \sum_k c_k\ \psi_k^{(j)}\right) \\
&= \sum_i c_i \sum_k c_k\ \left(\psi_i^{(j)}, \psi_k^{(j)}\right) \\
&= \sum_i c_i\ (c_i) \\
&= \sum_i c_i^2 \\
&= |||v|||_0^2;
\end{aligned}
\tag{3.30}
$$

thus $|||v|||_0$ is comparable to $\|v\|_{L^2}$. Due to the ellipticity of the bilinear form a, we have

$$
c^{-1}\ |||v|||_1 \leq \|v\|_1 \leq\ c\ |||v|||_1 \quad [5]. \tag{3.31}
$$

Given the shape regularity and quasi-uniformity of the grids, it can then be shown using interpolation on $|||v|||_s$ for $s = 0, 1$ that

$$
h_j^{1-s}\ c^{-1}\ |||v|||_s \leq \|v\|_s \leq h_j^{1-s}\ c\ |||v|||_s. \tag{3.32}
$$

Thus

$$
|||v|||_{1-\alpha} \leq c\, h_j^{-\alpha} \|v\|_{1-\alpha}.
$$

Let $u \in H_0^1$ be the solution of the dual problem

$$
a(u,v) = (g,v) \quad \text{for all } v \in H_0^1, \tag{3.33}
$$

and let $z \in V_j$ be the solution of

$$a(z, v) = 0 \quad \text{for all } v \in V_j. \tag{3.34}$$

Then

$$\begin{aligned}
(g, v) &= a(u, v) && \text{by (3.33)} \\
&= a(u - z, v) + a(z, v) \\
&= a(u - z, v) && \text{because } a(z, v) = 0 \\
&\leq |||u - z|||\ |||v||| \\
&= |||u - z|||_1 |||v||| && (3.35) \\
&\leq c\, \|u - z\|_1 |||v||| && \text{by (3.32)} \\
&\leq c\, h_j^{1+\alpha} h^{-1} \|u\|_{1+\alpha} |||v||| && \text{by Theorem 3.6.3} \\
&= c\, h_j^{\alpha} \|u\|_{1+\alpha} |||v||| \\
&\leq c\, h_j^{\alpha} \|g\|_{1-\alpha} |||v|| && \text{by (3.26)} \quad \forall\, g \in H^{\alpha-1}
\end{aligned}$$

By an Aubin-Nitsche duality argument, we can express the norm in $H^{1-\alpha}$ as

$$\begin{aligned}
\|v\|_{1-\alpha} &= \sup_{g \in H^{1-\alpha}} \frac{(g, v)}{\|g\|_{1-\alpha}} && \text{where } g \neq 0 \\
&\leq \sup_{g \in H^{1-\alpha}} \frac{1}{\|g\|_{1-\alpha}}\, c\, h_j^{\alpha} \|g\|_{1-\alpha} |||v||| && (3.36) \\
&= c\, h_j^{\alpha} |||v|||,
\end{aligned}$$

and (3.28) follows.

Cascadic Algorithm

Let u_j denote the finite element approximation on level j and let S_{j,m_j} represent the iterative operator on level j for the symmetric Gauss-Seidel, SSOR, or damped Jacobi method with m_j iterations. On the coarsest grid a direct solve is performed on the finite element system (3.25). The cascadic multigrid method is given in Algorithm 3.5. Here u_j^0 is the initial approximation on level j, u_j^* is the iterative approximation,

Algorithm 3.5 Cascadic multigrid with general energy reducing smoother

1. If $j = 0$
   (a) Direct solve to get u_0
   (b) Set $u_0^* = u_0$
2. If $j > 0$
   (a) Interpolate u_{j-1} to get initial approximation: $u_j^0 = P u_{j-1}^*$
   (b) Perform m_j smoothing iterations: $u_j^* = S_{j,m_j} u_j^0$

and P is an interpolation operator. In the next section we suppose the smoother is the symmetric Gauss-Seidel iteration.

Symmetric Gauss-Seidel Iteration

Let u_j denote the finite element approximation on level j and let S_{j,m_j}^{SGS} represent the iterative operator on level j for the Symmetric Gauss-Seidel method with m_j iterations. On the coarsest grid a direct solve is performed on the finite element system (3.25). Thus the cascadic multigrid method is given by

$$\begin{cases} (i) & u_0^* & = & u_0 & \\ (ii) & u_j^0 & = & Pu_{j-1}^* & \text{for } j = 1, \dots, l \\ (iii) & u_j^* & = & S_{j,m_j}^{SGS} u_j^0 & \text{for } j = 1, \dots, 1. \end{cases} \tag{3.37}$$

As defined in [3], this cascadic multigrid method will be *optimal* on level j if both *accuracy in the energy norm*

$$|||u_j - u_j^*||| \approx |||u - u_j|||$$

(the iterative error is comparable to the approximation error) and *multigrid complexity*

$$\text{amount of work} = \mathcal{O}(n_j),$$

where n_j=dimV_j, are obtainable simultaneously.

Let the space V_j have a basis $\Phi_k^{(j)}$ and define $A_{ik} = a(\Phi_i^{(j)}, \Phi_k^{(j)})$ and $b_i = b(\Phi_i^{(j)})$. Then (3.25) can be written as $Au_j = b$, where $u_j = \sum_{k=1}^{n_j} u_k^{(j)} \Phi_k^{(j)}$.

As the Symmetric Gauss-Seidel iteration is a linear function of u_j and b, it can be represented as $S_{j,1}^{SGS} u_j = M_j u_j + N_j b$. Then

$$u_j^{k+1} = S_{j,1}^{SGS} u_j^k = M_j u_j + N_j b \tag{3.38}$$

and the k^{th} iterate u_j^k can be expressed as

$$u_j^k = S_{j,k}^{SGS} u_j^0 = M_j^k u_j^0 + N_j^{(k)} b, \tag{3.39}$$

where $N_j^{(k)} = \sum_{i=0}^{k-1} M_j^i N_j$. The finite element solution u_j on level j is a fixed point of (3.38); thus $u_j = M_j u_j + N_j b$, or $I = M_j + N_j A$, where I is the identity matrix. If A is non-singular, then the matrix N_j can be written as $N_j = (I - M_j)A^{-1}$. Then the error of the $(k+1)^{st}$ iterate u_j^{k+1} is given by

$$\begin{aligned} u_j^{k+1} &= S_{j,k+1}^{SGS} u_j^0 - u_j \\ &= M_j u_j^k + N_j b - u_j \\ &= M_j u_j^k + (I - M_j)A^{-1} b - u_j \\ &= M_j u_j^k + (I - M_j) u_j - u_j \\ &= M_j u_j^k + u_j - M_j u_j - u_j \\ &= M_j (u_j^k - u_j). \end{aligned} \tag{3.40}$$

The error $u_j^k - u_j = S_{j,k}^{SGS} u_j^0 - u_j$ can be represented as

$$\begin{aligned} u_j^k - u_j &= M_j^k u_j^0 + N_j^{(k)} b - u_j \\ &= M_j^k u_j^0 + (\sum_{i=0}^{k-1} M_j^i N_j) b - u_j \\ &= M_j^k u_j^0 + \sum_{i=0}^{k-1} M_j^i (I - M_j) A^{-1} b - u_j \\ &= M_j^k u_j^0 + \sum_{i=0}^{k-1} (M_j^i - M_j^{i+1}) u_j - u_j \\ &= M_j^k u_j^0 + (M_j^0 - M_j^k) u_j - u_j \\ &= M_j^k (u_j^0 - u_j). \end{aligned} \tag{3.41}$$

Equations (3.40) and (3.41) have been expanded from [14].

Smoothing Properties

Define a matrix norm as follows:

$$\|A\|_{F \leftarrow V} := \sup_{0 \neq u \in V} \frac{\|Au\|_F}{\|u\|_V},$$

where $\|\cdot\|_F$ and $\|\cdot\|_V$ are the norms on spaces F and V, respectively. Similarly, define $|||A||| := \|A\|_{a \leftarrow a} = \sup_{u \neq 0} \dfrac{|||Au|||}{|||u|||}$ and $\|A\|_{a \leftarrow 0} = \dfrac{|||A|||}{\|u\|_0}$, where $\|\cdot\|_0 \equiv \|\cdot\|_{L^2}$.

For given norms $\|\cdot\|_F$ and $\|\cdot\|_V$, we can define $\mathfrak{S}_{j,m_j}^{SGS}$ as having the *smoothing* *property* [14] if there exist functions $\eta(m_j)$ and $\overline{m_j}(h)$ and a number α such that

$$\begin{aligned}
&\|A_j \mathfrak{S}^{SGS}_{j,m_j}\|_{F\leftarrow V} \le \eta(m_j) h_j^{-\alpha} \text{ for all } 1 \le m_j \le \overline{m_j}(h), j \ge 1,\\
&\eta(m_j) \to 0 \text{ as } m_j \to \infty, \qquad (3.42)\\
&\overline{m_j}(h) = \infty \text{ or } \overline{m_j}(h) \to \infty \text{ as } h \to 0.
\end{aligned}$$

To show that $\mathfrak{S}^{SGS}_{j,m_j}$ has the smoothing property, $A_j \mathfrak{S}^{SGS}_{j,m_j}$ can be expressed in the form $B(I-B)^{m_j}$ for positive definite B with $0 \le B \le I$. For a matrix B with $B = B^*$,

$$\|B(I-B)^k\|_{V\leftarrow V} = \max\{|f(\lambda)| : \lambda \in \sigma(B)\} \le \eta_0(k) \qquad (3.43)$$

where $f(x) = x(1-x)^k$ and $\eta_0(k) := \frac{k^k}{(k+1)^{k+1}}$. [14] For $k \ge 1$, $\eta_0(k) \le \frac{1}{k}$.

As described in the Algorithms chapter, the Symmetric Gauss-Seidel iteration can be expressed as

$$\begin{gathered}
\text{For } i = 1, 2, \ldots, n_j - 1, n_j, n_j - 1, \ldots, 1\\
u_{j,i} = \left[\sum_{k=1,k\ne i}^{n_j} A_{j,ik} u_{j,k} - b_{j,i}\right] / A_{j,ii}
\end{gathered} \qquad (3.44)$$

In matrix form, the iteration 3.44 can be expressed as

$$\begin{aligned}
\hat{u}_j^{k+1} &= (D_j - L_j)^{-1}(U_j u_j^k + b_j)\\
u_j^{k+1} &= (D_j - U_j)^{-1}(L_j \hat{u}_j^{j+1} + b_j),
\end{aligned} \qquad (3.45)$$

where L_j and U_j are strictly lower and upper triangular matrices, respectively, and D_j is the diagonal matrix such that $A_j = D_j - L_j - U_j$. Thus the iteration matrix can be written as

$$
\begin{aligned}
M_j &= (D_j - U_j)^{-1} L_j (D_j - L_j)^{-1} U_j \\
&= I - [(D_j - L_j) D_j^{-1} (D_j - U_j)]^{-1} A_j
\end{aligned}
\tag{3.46}
$$

The Symmetric Gauss-Seidel iteration can be written as

$$
W_j(u_j^k - u_j^{k+1}) = A_j u_j^k - b_j, \tag{3.47}
$$

where $W_j = (D_j - L_j) D_j^{-1} (D_j - U_j)$. If $A_j = A_j^* > 0$ and $D_j = D_j^* > 0$, then W_j is symmetric positive definite and satisfies $0 < A_j \leq W_j$.

Define $\|u_j\|_V := |A_j^{\frac{1}{2}} u_j|_0$, the discrete energy norm, $\|u_j\|_F := |W_j^{-\frac{1}{2}} u_j|_0$ and $0 < V := A_j^{\frac{1}{2}} W_j^{-1} A^{\frac{1}{2}} \leq I$. Then, by (3.43),

$$
\begin{aligned}
\|A_j \mathfrak{S}_{j,m_j}^{SGS}\|_{F\leftarrow V} &= \|A_j(I-[(D_j-L_j)D_j^{-1}(D_j-U_j)]^{-1}A_j)^{m_j}\|_{F\leftarrow V} \\
&= \|A_j(I-W_j^{-1}A_j)^{m_j}\|_{F\leftarrow V} \\
&= \sup \tfrac{\|A_j(I-W_j^{-1}A_j)^{m_j}u\|_F}{\|u\|_V} \\
&= \sup \tfrac{|W_j^{-\frac{1}{2}}A_j(I-W_j^{-1}A_j)^{m_j}u|_0}{|A_j^{\frac{1}{2}}u|_0} \\
&= \sup \tfrac{|W_j^{-\frac{1}{2}}A_j(I-W_j^{-1}A_j)^{m_j}A_j^{-\frac{1}{2}}A_j^{\frac{1}{2}}u|_0}{|A_j^{\frac{1}{2}}u|_0} \\
&= |||W_j^{-\frac{1}{2}}A_j(I-W_j^{-1}A_j)^{m_j}A^{-\frac{1}{2}}||| \\
&= |||W_j^{-\frac{1}{2}}A_j^{\frac{1}{2}}(I-A^{\frac{1}{2}}W_j^{-1}A_j^{\frac{1}{2}})^{m_j}||| \\
&= |||W_j^{-\frac{1}{2}}A_j^{\frac{1}{2}}(I-Y)^{m_j}||| \\
&= |||Y^{\frac{1}{2}}(I-Y)^{m_j}||| \\
&\leq \sqrt{\eta_0(2m_j)} \\
&< \sqrt{\tfrac{1}{2m_j}} \\
&\leq \tfrac{c}{m_j^{\frac{1}{2}}}
\end{aligned}
\tag{3.48}
$$

The Symmetric Gauss-Seidel iteration is an *energy reducing smoother* [3] since the following holds for all $v_j \in V_j$:

$$
\begin{aligned}
(i) \quad |||\mathfrak{S}_{j,m_j}^{SGS} v_j||| &\leq c\,\frac{h_j^{-1}}{m_j^{1/2}}\|v_j\|_0 \\
(ii) \quad |||\mathfrak{S}_{j,m_j}^{SGS} v_j||| &\leq |||v_j|||
\end{aligned}
\tag{3.49}
$$

Proof:

Equation (3.49) (ii) can be shown as follows:

$$\begin{aligned} |||\mathfrak{S}^{SGS}_{j,m_j} v_j||| &\leq |||\mathfrak{S}^{SGS}_{j,m_j}|||\ |||v_j||| \\ &\leq \|A_j^{-1}\|_{V\leftarrow F}\ \|A_j \mathfrak{S}^{SGS}_{j,m_j}\|_{F\leftarrow V}\ \|v_j\| \\ &\leq \sqrt{\eta_0(2m_j)}\ |||v_j||| \leq |||v_j|||. \end{aligned} \tag{3.50}$$

■

Comparing the L^2 and energy norms on the right hand side in (3.49) with discrete norms defined in [2], we can bound the left hand norm by the $H^{1-\alpha}$-norm as follows.

Lemma 3.3.2 [3] *A smoother in the sense of (3.49) fulfills*

$$|||\mathfrak{S}^{SGS}_{j,m_j} v_j||| \leq c \frac{h_j^{-\alpha}}{m_j^{\alpha/2}} \|v_j\|_{1-\alpha} \tag{3.51}$$

Proof:

This can be shown using interpolation and duality arguments similar to those given for (3.28).

■

For the cascadic multigrid method given in Algorithm 3.5, it can be shown that

Theorem 3.3.3 holds.

Theorem 3.3.3 [3] *The error of the cascadic multigrid method with an energy reducing smoother can be estimated by*

$$|||u_l - u_l^*||| \le c \sum_{j=1}^{l} \frac{1}{m_j^{\alpha\gamma}} |||u_j - u_{j-1}||| \le c \sum_{j=1}^{l} \frac{h_j^\alpha}{m_j^{\alpha\gamma}} \|f\|_{H^{\alpha-1}}. \tag{3.52}$$

PROOF:

For $j = 1, \ldots, l$

$$\begin{aligned} |||u_j - u_j^*||| &= |||u_j - S_{j,m_j} u_j^*||| && (u_j^* = S_{j,m_j} u_{j-1}^*) \\ &= |||\mathfrak{S}_{j,m_j}(u_j - u_{j-1}^*)||| && (u_j - S_{j,m_j} u_j^0 = \mathfrak{S}_{j,m_j}(u_j - u_j^0)) \\ &\le |||\mathfrak{S}_{j,m_j}(u_j - u_{j-1})||| \\ &\quad + |||\mathfrak{S}_{j,m_j}(u_{j-1} - u_{j-1}^*)||| && \text{by the linearity of } \mathfrak{S}_{j,m_j}. \end{aligned} \tag{3.53}$$

<u>First Term:</u>

$$\begin{aligned} |||\mathfrak{S}_{j,m_j}(u_j - u_{j-1})||| &\le c \frac{h_j^{-\alpha}}{m_j^{\alpha\gamma}} \|u_j - u_{j-1}\|_{H^{1-\alpha}} && \text{by Lemma 3.3.2} \\ &\le c \frac{1}{m_j^{\alpha\gamma}} |||u_j - u_{j-1}||| \end{aligned} \tag{3.54}$$

This is by the approximation property:

$$\|u_j - u_{j-1}\|_{H^{1-\alpha}} \;\leq\; c\,h_j^{\alpha}|||u_j - u_{j-1}|||, \text{ for } j = 1, \ldots, l$$

.

<u>Second Term:</u> By Property (ii) of a smoother ($|||\mathfrak{S}_{j,m_j} v_j||| \;\leq\; |||v_j|||$), we have $|||\mathfrak{S}_{j,m_j}(u_{j-1} - u^*_{j-1})||| \leq |||u_{j-1} - u^*_{j-1}|||$. Thus

$$\|u_j - u^*_j||| \;\leq\; \frac{c}{m_j^{\alpha\gamma}}|||u_j - u_{j-1}||| + |||u_{j-1} - u^*_{j-1}|||.$$

Using $u^*_o = u_0$, (by induction):

$$
\begin{aligned}
|||u_1 - u_1^*||| &\leq \frac{c}{m_1^{\alpha\gamma}} |||u_1 - u_0||| + |||u_0 - u_0||| \text{ (Note that the last term is 0.)} \\
|||u_2 - u_2^*||| &\leq \frac{c}{m_2^{\alpha\gamma}} |||u_2 - u_1||| + |||u_1 - u_1^*||| \\
&\leq \frac{c}{m_1^{\alpha\gamma}} |||u_2 - u_1||| + \frac{c}{m_1^{\alpha\gamma}} |||u_1 - u_0||| \\
&\vdots \\
|||u_l - u_l^*||| &\leq c \sum_{j=1}^{l} \frac{1}{m_j^{\alpha\gamma}} |||u_j - u_{j-1}||| \\
&\leq c \sum_{j=1}^{l} \frac{1}{m_j^{\alpha\gamma}} |||u - u_{j-1}||| \text{ by Galerkin orthogonality} \\
&\leq c \sum_{j=1}^{l} \frac{h_j^{\alpha}}{m_j^{\alpha\gamma}} ||b||_{H^{\alpha-1}}
\end{aligned}
\tag{3.55}
$$

■

The last term comes from the approximation error of the finite element method in energy norm.

Since

$$
\frac{1}{c} 2^{l-j} h_l \leq h_j \leq c\, 2^{l-j} h_l, \tag{3.56}
$$

we consider sequences $m_1, \ldots, m_l$ of the kind

$$m_j = \lceil \beta^{l-j} m_l \rceil \tag{3.57}$$

for some fixed $\beta > 1$. This ensures that the number of iterations decreases as the solution is approximated on finer meshes.

For large β, the accuracy of the cascadic multigrid method is given in the following lemma.

Lemma 3.3.4 [3] *Let the number m_j of iterations on level j be given by (3.57). The error of the cascadic multigrid method is*

$$|||u_l - u_l^*||| \leq \begin{cases} \dfrac{c}{1-(2/\beta^\gamma)^\alpha} \dfrac{h_l^\alpha}{m_l^{\alpha\gamma}} \|b\|_{H^{\alpha-1}}, & \text{for } \beta > 2^{1/\gamma}, \\[2ex] c\, l\, \dfrac{h_l^\alpha}{m_l^{\alpha\gamma}} \|b\|_{H^{\alpha-1}}, & \text{for } \beta = 2^{1/\gamma}. \end{cases} \tag{3.58}$$

PROOF:

This proof, given in [3], has been expanded here to show more detail.

By Theorem 3.3.3:

$$\|u_l - u_l^*\||\ \ \leq\ \ c \sum_{j=1}^{l} \frac{h_j^\alpha}{m_j^{\alpha\gamma}} \|b\|_{H^{\alpha-1}} \tag{3.59}$$

$$\frac{2^{l-j}h_l}{c} \;\leq\; h_j \;\leq\; c\,2^{l-j}h_j \tag{3.60}$$

$$\begin{aligned}
c\sum_{j=1}^{l}\frac{h_j^\alpha}{m_j^{\alpha\gamma}}\|b\|_{H^{\alpha-1}} &= c\,\|b\|_{H^{\alpha-1}}\sum_{j=1}^{l}\frac{h_j^\alpha}{m_j^\alpha} \\
&\leq c\,\|b\|_{H^{\alpha-1}}\sum_{j=1}^{l}\frac{\left(2^{l-j}h_l\right)^\alpha}{\left(\beta^{l-j}m_l\right)^{\alpha\gamma}} \\
&\quad \left(\begin{array}{lcl} h_j & \leq & c2^{l-j}h_l \\ m_j & \geq & \beta^{l-j}m_l \end{array}\right) \\
&= c\,\|b\|_{H^{\alpha-1}}\sum_{j=1}^{l}\left(\tfrac{2}{\beta^\gamma}\right)^{\alpha(l-j)}\cdot\frac{h_l^\alpha}{m_l^{\alpha\gamma}} \qquad (3.61) \\
&= c\,\|b\|_{H^{\alpha-1}}\frac{h_l^\alpha}{m_l^{\alpha\gamma}}\sum_{j=0}^{l-1}\left(\tfrac{2}{\beta^\gamma}\right)^{\alpha j} \\
&\quad \begin{cases} \leq \; c\,\|b\|_{H^{\alpha-1}}\dfrac{h_l^\alpha}{m_l^{\alpha\gamma}}\cdot\dfrac{1}{1-\left(\frac{2}{\beta^\gamma}\right)^\alpha} & \text{if } \beta^\gamma > 2 \\[2ex] = \; c\cdot l\cdot\dfrac{h_l^\alpha}{m_l^{\alpha\gamma}}\cdot\|b\|_{H^{\alpha-1}} & \text{if } \beta^\gamma = 2 \end{cases}
\end{aligned}$$

■

For small β, the optimal complexity of the method is given in Lemma 3.3.5.

Lemma 3.3.5 [3] *If the number m_j of iterations on level j is given by (3.57), then* *the computational cost of the cascadic multigrid method is proportional to*

$$\sum_{j=1}^{l} m_j n_j \leq \begin{cases} \dfrac{c}{1-\beta/2^d}\, m_l\, n_l, & \textit{for } \beta < 2^d, \\ c\, l\, m_l\, n_l, & \textit{for } \beta = 2^d. \end{cases} \tag{3.62}$$

PROOF:

This proof, given in [3], has also been expanded to show more detail.

$$\frac{2^{dj}}{c} \leq n_j = \mathrm{dim} V_j \leq c\, 2^{dj}.$$

$$m_j \approx \beta^{l-j} m_j,$$

$$n_{j-1} \approx \frac{n_j}{2^d} \leq c \frac{2^{dj}}{2^d},$$

$$n_l \approx 2^d n_{l-1} \approx (2^d)^{l-j} n_j.$$

$$\begin{aligned}
\sum_{j=1}^{l} m_j n_j &\approx \sum_{j=1}^{l} \beta^{l-j} m_l n_j \\
&= m_l \sum_{j=1}^{l} \beta^{l-j} n_j \\
&\approx m_l \sum_{j=1}^{l} \beta^{l-j} \cdot \tfrac{1}{(2^d)^{l-j}} \cdot n_l \qquad (3.63)\\
&= m_l n_l \sum_{j=1}^{l} \left(\tfrac{\beta}{2^d}\right)^{l-j} \\
&= m_l n_l \sum_{j=0}^{l-1} \left(\tfrac{\beta}{2^d}\right)^{j}.
\end{aligned}$$

If $\beta < 2^d$,

$$\sum_{j=0}^{l-1} \left(\frac{\beta}{2^d}\right)^j = \frac{1-\left(\frac{\beta}{2^d}\right)^l}{1-\frac{\beta}{2^d}} < \frac{1}{1-\frac{\beta}{2^d}}.$$

If $\beta = 2^d$,

$$\sum_{j=0}^{l-1} \left(\frac{\beta}{2^d}\right)^j = \sum_{j=0}^{l-1} 1 = l.$$

■

Combining Lemmas 3.3.4 and 3.3.5, we see that the method will be optimal if β satisfies $2^{1/\gamma} < \beta < 2^d$; thus γ must be larger than d^{-1}. If $\gamma = d^{-1}$, optimality can only be obtained in either the accuracy or the complexity.

For example,

Lemma 3.3.6 [3] *Let $\gamma = 1/d$. Choose $\beta = 2^d$ (thus complexity will deteriorate) and the number of iterations on level l as $m_l = \lceil m_* \cdot l^{d/\alpha} \rceil$. Then the error of the cascadic* *multigrid method is*

$$|||u_l - u_l^*||| \leq c \frac{h_l^\alpha}{m_*^{\alpha/d}} \|b\|_{H^{\alpha-1}}, \tag{3.64}$$

and the complexity is

$$\sum_{j=1}^{l} m_j n_j \leq c m_* n_l \left(1 + \log n_l\right)^{1+d/\alpha}. \tag{3.65}$$

PROOF:

The proof of Lemma 3.3.6 was omitted from [3]. I have constructed the proof as follows.

$$\gamma = \frac{1}{d}, \beta = 2^d = 2^{\frac{1}{\gamma}}.$$

$$
\begin{aligned}
|||u_l - u_l^*||| &\leq c \cdot l \cdot \frac{h_l^\alpha}{m_l^{\alpha\gamma}} \|b\|_{H^{\alpha-1}} && \text{(by Lemma 3.3.4)} \\
&\leq c \cdot l \cdot \frac{h_l^\alpha}{\left(m_* l^{d/\alpha}\right)^{\alpha\gamma}} \|b\|_{H^{\alpha-1}} && \text{because } m_l \geq m_* \cdot l^{d/\alpha} \\
&= c \cdot l \cdot \frac{h_l^\alpha}{m_*^{\alpha\gamma} l^{d\gamma}} \|b\|_{H^{\alpha-1}} \\
&= c \cdot l \cdot \frac{h_l^\alpha}{m_*^{\alpha\gamma} l} \|b\|_{H^{\alpha-1}} \qquad (3.66) \\
&= c \cdot \frac{h_l^\alpha}{m_*^{\alpha\gamma}} \|b\|_{H^{\alpha-1}} \\
&= c \frac{h_l^\alpha}{m_*^{\alpha/d}} \|b\|_{H^{\alpha-1}} \\
\textstyle\sum_{j=1}^{l} m_j n_j &\leq c \cdot l \cdot m_l \cdot n_l && \text{(by Lemma 3.3.5)} \\
&\approx c \cdot l \cdot n_l \cdot m_* \cdot l^{d/\alpha} \\
&= c \cdot m_* \cdot n_l \cdot \left(l^{1+d/\alpha}\right)
\end{aligned}
$$

$$\begin{aligned}
\frac{2^{dj}}{c} &\leq \quad n_j \quad \leq c\,2^{dj} \text{ thus,} \\
\frac{2^{dl}}{c} &\leq \quad n_l \quad \leq c\,2^{dl} \qquad (3.67) \\
\log\frac{2^{dl}}{c} &\leq \log n_l \leq \log\left(c\,2^{dl}\right) \\
l\log 2^d - \log c &\leq \log n_l \leq l\log 2^d + \log c
\end{aligned}$$

$$\begin{aligned}
l\log 2^d &\leq \log n_l + \log c \\
l &\leq \frac{1}{\log 2^d}(\log n_l + \log c) \qquad (3.68) \\
&= \tilde{c}\,(\log n_l + \log c) \\
&\leq \tilde{c}\,(\log n_l + 1)
\end{aligned}$$

Thus

$$\sum_{j=1}^{l} \leq c \cdot m_* \cdot n_l \cdot (\log n_l + 1)^{1+d/\alpha}.$$

■

As mentioned at the beginning of this section, $\gamma = \frac{1}{2}$ for the symmetric Gauss-Seidel, SSOR, or damped Jacobi iterations. Thus, for one of these iterations, the cascadic multigrid method yields the following errors and complexities:

In two dimensions, $d = 2$, so $\gamma = \frac{1}{d}$. The error is then given by

$$|||u_l - u_l^*||| \leq c \frac{h_l^\alpha}{m_*^{\alpha/2}} ||b||_{H^{\alpha-1}}, \tag{3.69}$$

where $m_l = \lceil m_* \cdot l^{2/\alpha} \rceil$, and the complexity is

$$\sum_{j=1}^{l} m_j n_j \leq c\, m_* n_l \left(1 + \log n_l\right)^{1+2/\alpha}. \tag{3.70}$$

Thus in two dimensions, either the accuracy or the complexity deteriorates for the symmetric Gauss-Seidel, SSOR, or damped Jacobi.

In three dimensions, $d = 3$, so $\gamma > \frac{1}{d}$ and $4 < \beta < 8$. The error is

$$|||u_l - u_l^*||| \leq c \frac{1}{1 - \left(2/\beta^{\frac{1}{2}}\right)^\alpha} \cdot \frac{h_l^\alpha}{m_l^{\alpha/2}} ||b||_{H^{\alpha-1}}, \tag{3.71}$$

and the complexity is

$$\sum_{j=1}^{l} m_j n_j \leq c \cdot \frac{1}{1 - \beta/8} \cdot m_l \cdot n_l. \tag{3.72}$$

Thus the cascadic multigrid method in three dimensions is optimal.

Note: Bornemann, Deuflhard and Shaidurov proved multigrid optimality with respect to the energy norm. Bornemann and Krause [4] then showed that cascadic multigrid cannot be optimal with respect to the L^2 norm for linear finite elements.

However, they argue that higher order finite elements allow optimality.

3.4 Other conforming and nonconforming elements

The cascadic multigrid method can also be used for elements other than the P1 conforming element. The description of such problems appears at the end of this section.

For these problems, we may not have nested finite element spaces. Thus we will use an interpolator P_j from the finite element space V_{j-1} to V_j. Set $t = 1$ for the second order problem and $t = 2$ for the fourth order problem. Then the interpolator is defined as follows:

$$\begin{aligned} (i) \qquad |v - P_j v|_{t-1,j} &\leq Ch_j |v|_{t,j-1} \quad \forall v \in V_{j-1}, \\ (ii) \quad |u_j - P_j u_{j-1}|_{t-1,l} &\leq Ch_j^2 \|b\|_0, \end{aligned} \tag{3.73}$$

where u_j is the finite element solution on V_j. The semi-norm $|\cdot|_{t,j}$ is a norm on V_j defined by

$$|v|_{i,j}^2 = \sum_{T \in \mathcal{T}_j} |v|_{i,T}^2, \tag{3.74}$$

for $i = 0, \ldots, t$, where $|v|_{i,T}^2$ is the semi-norm in the Sobolev space on T. The norm

$|\cdot|_{t,j}$ satisfies the following:

$$
\begin{aligned}
|\cdot|_{0,j} &\approx \|\cdot\|_0, \text{ and} \\
c_1|v|_{i,j} \le |||v|||_j &\le c_2|v|_{i,j},
\end{aligned}
\tag{3.75}
$$

where $|||v|||_j$ is the energy norm on level j.

Define S_{j,m_j} as the smoother and $\mathfrak{S}$ as the linear operator satisfying both (3.18) and the following:

$$
\begin{aligned}
(i) \quad |||\mathfrak{S}_j^{m_j} v|||_j &\le c\frac{h_j^{-1}}{m_j^{\gamma}}|||v|||_j, \quad \forall\, v \in V_j, \\
(ii) \quad |||\mathfrak{S}_j^{m_j} v|||_j &\le |||v|||_j.
\end{aligned}
\tag{3.76}
$$

That is, S_{j,m_j} satisfies the smoothing properties (3.76).

Finally, define the projection operator $R_j : V_{j-1} + V_j \to V_j$ such that

$$a_j(R_j u, v) = a_j(u, v), \tag{3.77}$$

for all $v \in V_j$, and

$$|v - R_j v|_{t-1,j} \le c\, h_j\, |v|_{t,j-1}, \tag{3.78}$$

for all $v \in V_{j-1}$.

The following is a more precise statement of Lemma 2.1 from [22].

Lemma 3.4.1 *If (3.73), (3.76), and (3.78) hold and m_j satisfies (3.57), then*

$$|||u_l - u_l^*|||_l \le c\sum_{j=1}^{l} \frac{h_j}{m_j^{\gamma}} \|b\|_0. \tag{3.79}$$

Note that this is Theorem 3.3.3 with $\alpha = 1$.

3.4.1 Second order problem

The problem to be solved is the second order elliptic problem

$$\begin{aligned} -\sum_{i,j=1}^{d} \frac{\partial}{\partial x_i}(a_{ij}\frac{\partial u}{\partial x_j}) &= b \quad \text{in } \Omega, \\ u &= 0 \quad \text{on } \partial\Omega, \end{aligned} \tag{3.80}$$

where $\Omega \subset \mathbb{R}^d$, for $d = 2, 3$, is a convex polygon and $b \in L^2(\Omega)$. The functions a_{ij} and b are assumed to be relatively smooth. The solution u satisfies H^2-regularity:

$$||u||_2 \leq c||b||_0. \tag{3.81}$$

The bilinear form is

$$a(u, v) = \int_\Omega \sum_{i,j=1}^{d} a_{ij} \frac{\partial u}{\partial x_j} \frac{\partial v}{\partial x_i} dx, \tag{3.82}$$

for all $u, v \in H^1$, and the discrete version is

$$a_j(u, v) = \sum_{T \in \mathcal{T}_j} \int_T \sum_{i,j=1}^{d} a_{ij} \frac{\partial u}{\partial x_j} \frac{\partial v}{\partial x_i} dx. \tag{3.83}$$

For large β, the accuracy of the cascadic multigrid method is given in the following lemma.

Lemma 3.4.2 [22] *Let the number m_j of iterations on level j be given by (3.57). The* *error of the cascadic multigrid method is*

$$|||u_l - u_l^*||| \leq \begin{cases} \dfrac{c}{1-(2/\beta^\gamma)} \dfrac{h_l}{m_l^\gamma} ||b||_0, & \textit{for } \beta > 2^{1/\gamma}, \\[2ex] c\, l\, \dfrac{h_l}{m_l^\gamma} ||b||_0, & \textit{for } \beta = 2^{1/\gamma}. \end{cases} \tag{3.84}$$

For small β, the optimal complexity of the method is given in Lemma 3.4.3.

Lemma 3.4.3 [22] *Under the assumptions of Lemma 3.4.2, the computational cost of the cascadic multigrid method is proportional to*

$$\sum_{j=0}^{l} m_j n_j \leq \begin{cases} \dfrac{c}{1-\beta/2^d}\, m_l\, n_l, & \textit{for } \beta < 2^d, \\[2ex] c\, l\, m_l\, n_l, & \textit{for } \beta = 2^d. \end{cases} \tag{3.85}$$

Using Lemmas 3.4.2 and 3.4.3, we have an optimal method if $2^{1/\gamma} < \beta < 2^d$.

Theorem 3.4.4 [22] *Suppose (3.73), (3.76), and (3.78) hold.*

1. *The cascadic multigrid in three dimensions will be optimal for the symmetric Gauss-Seidel, SSOR, and damped Jacobi iterations. In this case,* $d = 3$ *and* $\gamma = \frac{1}{2}$*, thus we can choose* $\beta \in (4, 8)$.

2. *In two dimensions, the cascadic multigrid will be accurate with complexity* $\mathcal{O}(n_l(1 + \log n_l)^3)$ *for the energy reducing smoothers.*

3. For the conjugate gradient iteration, $\gamma = 1$, thus the cascadic method will be optimal in both two and three dimensions.

Lagrange conforming elements

The Lagrange element is the most widely used element [7]. The P1 conforming element is the linear Lagrange element. The quadratic Lagrange element is used for polynomials of degree ≤ 2, defined at the vertices of the triangle as well as at the midpoints of each edge. The cubic Lagrange uses polynomials defined at the vertices, the trisections of the edges, and the centroid of the triangle. In general, for $k \geq 1$, the Lagrange element uses the 3 vertices, $3(k-1)$ edge nodes, and $\frac{1}{2}(k-2)(k-1)$ points on the interior of the element.

In the case of Lagrange conforming elements, the finite element spaces are nested. Thus the interpolation operator P_j and the projection operator R_j are equal to the identity I. Thus (3.73) holds for the second order problem ($t = 1$) with Lagrange conforming elements:

$$\begin{aligned} &(i) \quad 0 \leq Ch_j |v|_{t,j-1} \quad \forall v \in V_{j-1}, \\ &(ii) \quad \|u_j - u_{j-1}\|_0 \leq \|u_j - u\|_0 + \|u - u_{j-1}\|_0 \leq Ch_j^2 \|b\|_0. \end{aligned} \tag{3.86}$$

Note that item (ii) in (3.86) is also (3.78) with $t = 1$ and $R_j = I$. The Jacobi and Gauss-Seidel iterations satisfy (3.76) with $\gamma = \frac{1}{2}$ and the conjugate gradient iteration satisfies (3.76) with $\gamma = 1$; thus Theorem 3.4.4 holds for the second order problem with Lagrange conforming elements.

Hermite elements

Cubic triangular Hermite elements fix the values of the polynomial and its first derivatives at the vertices of a triangle and the value at the center of the triangle. In this case, the finite element spaces are nonnested, so the interpolator cannot be the identity. Define P_j as follows:

$$\begin{aligned}
&(i) \quad P_j v = \tfrac{\partial P_j v}{\partial x_k} = 0, \text{ for k=1,2 at all vertices of } \mathcal{T}_j \text{ on } \partial\Omega, \\
&(ii) \quad P_j v(a_i) = v(a_i), P_j v(a_{ij}) = v(a_{ij}), P_j v(c) = v(c), \\
&(iii) \quad \tfrac{\partial P_j v}{\partial x_k}(a_i) = \tfrac{\partial v}{\partial x_k} v(a_i), \tfrac{\partial P_j v}{\partial x_k}(a_{ij}) = \tfrac{\partial v}{\partial x_k} v(a_{ij}), \\
&\qquad \tfrac{\partial P_j v}{\partial x_k}(c) = \tfrac{\partial v}{\partial x_k} v(c), \text{ for } k = 1, 2,
\end{aligned} \tag{3.87}$$

where a_i, $i = 1, 2, 3$, is a vertex of $T \in \mathcal{T}_j$, a_{ij} is the midpoint of the edge $\overline{a_i a_j}$, c is the center of T, and w^P is the linear interpolation of w on $\mathcal{T}_{j-1}$.

Lemma 3.4.5 *[22] (3.73) and (3.78) hold for Hermite elements.*

Lemma 3.4.6 *[22]*

1. *If symmetric Gauss-Seidel, SSOR, or damped Jacobi (or other standard smoother) is used, then (3.76) holds with $\gamma = \frac{1}{2}$.*

2. *If the conjugate gradient method is used, (3.76) holds with $\gamma = 1$.*

Thus the requirements of Theorem 3.4.4 are met for the cubic Hermite element.

P1 nonconforming element

The P1 nonconforming finite element space is defined to be the set of all functions v such that $v|_T$ is linear for all $T \in \mathcal{T}$, v is continuous at the midpoints of the edges of T and $v = 0$ on these midpoints. The interpolator P_j is defined as follows:

1. If the midpoint m of a side of $T \in \mathcal{T}_j$ lies in the interior of $T \in \mathcal{T}_{j-1}$, then $P_j(v)(m) = v(m)$.

2. If m lies on the common edge of two adjacent triangles $T_1, T_2 \in \mathcal{T}_{j-1}$, then $(P_j v)(m) = \frac{1}{2}(v|_{T_1}(m) + v|_{T_2}(m))$.

Lemma 3.4.7 *[22] (3.73) holds for the P1 nonconforming element.*

Lemma 3.4.6 holds for the P1 nonconforming element.

Lemma 3.4.8 *[22] (3.78) holds for the P1 nonconforming element.*

Thus Theorem 3.4.4 holds for the P1 nonconforming element.

Wilson nonconforming element

The Wilson element is defined on the square $0 \leq x, y \leq 1$ with six basis functions $xy, x(1-y), y(1-x), (1-x)(1-y), 4x(1-x)$ and $4y(1-y)$. Let V_j be the finite element space. Define U_j as the bilinear conforming counterpart to V_j and let π_j be the bilinear interpolation operator. Then

$$\pi_{j-1} v \in U_{j-1} \subset U_j \subset V_j; \tag{3.88}$$

thus $P_j : V_{j-1} \to V_j$ can be used for π_j. Equations (3.73), (3.76), and (3.78) hold for the Wilson element; thus Theorem 3.4.4 holds as well.[22]

3.4.2 Plate bending problem

The plate bending problem is defined as follows [7]: Let $a(\cdot,\cdot)$ be a bilinear form on $H^2(\Omega)$ such that

$$a(u,v) = \int_\Omega \Delta u \Delta v - (1-\nu)(2\frac{\partial^2 u}{\partial x^2}\frac{\partial^2 v}{\partial y^2} + 2\frac{\partial^2 u}{\partial y^2}\frac{\partial^2 v}{\partial x^2} - 4\frac{\partial^2 u}{\partial x \partial y}\frac{\partial^2 v}{\partial x \partial y})dxdy, \quad (3.89)$$

with $\nu \in [0, \frac{1}{2}]$.

The Morley finite element is used for the plate bending problem. The Morley finite element space is defined to be the set of functions $v|_T \in \mathcal{P}_2(T)$ such that v is continuous at the vertices of $T \in \mathcal{T}$ and the normal derivative of v is continuous at the midpoints of the edges of T. This element is nonnested; thus P_j is defined as

1. $P_j v = 0$ at each vertex of $\mathcal{T}_j$ along $\partial\Omega$, and $\frac{\partial P_j v}{\partial n} = 0$ at the midpoint of each edge along $\partial\Omega$.

2. If a vertex $p \in \mathcal{T}_j$ in the interior of Ω is also a vertex of $\mathcal{T}_{j-1}$, then $(P_j v)(p) = v(p)$. If p is the midpoint of a common edge of $T_1, T_2 \in \mathcal{T}_{j-1}$, then $(P_j v)(p) = \frac{1}{2}(v|_{T_1}(p) + v|_{T_2}(p))$.

3. If m is a midpoint of an edge in $\mathcal{T}_j$, n is the unit normal of the edge, and m is inside a triangle in $\mathcal{T}_{j-1}$, then $\frac{\partial P_j v}{\partial n}(m) = \frac{\partial v}{\partial n}(m)$. If the midpoint m is on the common edge of $T_1, T_2 \in \mathcal{T}_{j-1}$, then $\frac{\partial P_j v}{\partial n}(m) = \frac{1}{2}\left(\frac{\partial v|_{T_1}}{\partial n}(m) + \frac{\partial v|_{T_2}}{\partial n}(m)\right)$.

Conjugate gradients give only nearly optimal complexity, and other smoothers or roughers cannot be used at all.

3.5 Adaptive Cascadic Multigrid Method

The adaptive cascadic multigrid method can be used on nonuniform grids. The number of iterations on each level is determined by an a posteriori termination strategy based on the shape regularity of the triangulations [11].

In the algorithm, the stiffness matrix A_j is computed with respect to the scaled nodal basis $h_{j,i}^{(2-d)/2}\psi_{i,j}$, $i = 1, ..., n_j$, where $\{\psi_{j,i}\}_i$ is the usual nodal basis of the j^{th} linear finite element space and $h_{j,i} = \text{diam supp}\psi_{j,i}$. This scaling has the effect of diagonal preconditioning of the stiffness matrix. This scaling can eliminate problems resulting from discontinuous coefficients in two and three-dimensional problems.

Three variants of the adaptive cascadic multigrid were examined: the cascadic conjugate gradient method (CCG), the cascadic multigrid method with symmetric Gauss-Seidel (CSGS) seen in Algorithm 3.5, and the cascadic multigrid method with a multigrid V cycle as the basic iteration using a Jacobi smoother (Algorithm 3.6).

Algorithm 3.6 Cascadic multigrid with V cycle as basic iteration

1. If $j = 0$
   (a) Direct solve
2. If $j > 0$
   (a) Interpolate u_{j-1} to get initial approximation (not clear in paper that this is what happens)
   (b) Perform m_j multigrid V cycles using the Jacobi iteration

The method using the multigrid V cycle with adaptive grid refinement is the only method which is optimal with respect to the L^2-norm. The cascadic multigrid method is optimal with respect to the energy norm, but is not optimal in the L^2-norm.

3.6 Cascadic multigrid with change in order of discretization

New results are given this section, where a cascadic multigrid algorithm is implemented in which a higher order discretization is used on the coarser levels to accelerate the convergence. At a designated level k, $0 < k < l$, there is a switch to the standard (i.e., lower) method on the finer levels.

One reason for switching to a lower order method on the finer grids is that it is cheaper to use than the higher order method and cost can be a significant factor as the number of grid points gets larger. Not only is the smoother more expensive, but the interpolation process is, also.

Another reason is that, in a number of cases, a problem may arise which involves some post processing on the finest level that requires a particular order discretization. For example, in a combustion or computational fluid dynamics application, an auxiliary problem may be solved to recover the actual solution. If a fourth order method is used, problems may arise with continuity constraints on the data. For example, if there are sharp changes along a front, the high order method could result in a Gibb's effect, which completely destroys the value of the solution.

The notation used in Algorithms 3.7, 3.8, 3.9 and 3.10 is as follows:

- u the solution.
- u_j the initial approximation to the solution on level j.
- u_j^* the iterative approximation on level j.
- $E^{(j)}$ the energy norm on level j in the cascadic multigrid algorithms.
- E_2 the energy norm on level 2 computed in the nested iteration V cycle algorithm.

Note that while the algorithms here are specifically tailored to $\mathcal{O}(h^2)$ and $\mathcal{O}(h^4)$ discretizations, they can be generalized (and have been elsewhere) to any $\mathcal{O}(h^p)$ and $\mathcal{O}(h^q)$ methods, where $q > p$.

3.6.1 Two level V cycle

The energy norm of the error of a two level nested iteration V cycle using a second order discretization and interpolation is used to approximate the norm of the error on the finest grid l for a cascadic multigrid algorithm using the same discretization and interpolation. This norm will be used in the comparison of this second order algorithm with a cascadic algorithm using a fourth order discretization and interpolation on coarser levels which switches to second order on a particular fine mesh level $k+1$, where $0 < k+1 < l$.

If we assume that $h_j = \frac{h_{j-1}}{2}$, then the goal is to have the norm of the fourth order method on the finest level bounded by the V cycle energy norm divided by $2^{d(l-1)}$, where l is the number of levels and d is the dimension.

Algorithm 3.7 computes the energy norm of an approximate solution after a two level full multigrid cycle. The purpose of this algorithm is to estimate the error on some level $j > 2$. By having an estimate on level 2 and knowing that we have an $\mathcal{O}(h^2)$ discretization on levels 1, 2, and j, we only have to reduce the error on level j to approximately $(h_2/h_j)^{j-2}\|e_2\|$, where e_2 is the error on level 2.

Algorithm 3.7 Two level nested iteration V cycle to determine energy norm of $\mathcal{O}(h^2)$ method

1. Direct solve on level 0 to get u_0
2. Interpolate u_0 to get initial approximation $u_1 = P_0^{(2)} u_0$ on level 1
3. Perform a multigrid V cycle:
   (a) Smooth m times on u_1: $u_1^* = S_{1,1} u_1$. m is chosen using, e.g., Theorem 4 of [1].
   (b) Compute the residual $r_1 = b_1 - A_1 u_1^*$ and project to r_0 on level 0
   (c) Solve the residual problem $A_0 e = r_0$ with initial guess $e = 0$
   (d) Correct on level 1: $u_1 = u_1^* + e$ and smooth m times to get u_1^*
4. Compute energy norm $E_2 = \sqrt{(A_1 u_1^*, u_1^*) h^d}$ on level 1

Algorithm 3.7 uses a $\mathcal{O}(h^2)$ discretization (for example, see (2.14)) and linear interpolation, which injects the points from the coarse grid and determines the new points using averages of the surrounding points. For example, in two dimensions, the interpolation is given by

$$
\begin{aligned}
u^{(j)}_{2i,2k} &= u^{(j-1)}_{i,k} \\
u^{(j)}_{2i+1,2k} &= \tfrac{1}{2}(u^{(j-1)}_{i,k} + u^{(j-1)}_{i+1,k}) \\
u^{(j)}_{2i,2k+1} &= \tfrac{1}{2}(u^{(j-1)}_{i,k} + u^{(j-1)}_{i,k+1}) \\
u^{(j)}_{2i+1,2k+1} &= \tfrac{1}{4}(u^{(j-1)}_{i,k} + u^{(j-1)}_{i+1,k} + u^{(j-1)}_{i,k+1} + u^{(j-1)}_{i+1,k+1})
\end{aligned}
\tag{3.90}
$$

The projection used in the V cycle is the full weighted restriction $P_0^{(2)}$, given in (3.91) for the two dimensional restriction:

$$
\begin{aligned}
u^{(j-1)}_{i,k} = {} & \tfrac{1}{16}[u^{(j)}_{2i-1,2k-1} + u^{(j)}_{2i-1,2k+1} + u^{(j)}_{2i+1,2k-1} + u^{(j)}_{2i+1,2k+1} \\
& +2(u^{(j)}_{2i,2k-1} + u^{(j)}_{2i,2k+1} + u^{(j)}_{2i-1,2k} + u^{(j)}_{2i+1,2k}) + 4u^{(j)}_{2i,2k}].
\end{aligned}
\tag{3.91}
$$

The smoother is symmetric Gauss-Seidel, defined in (2.31).

The direct solve on the coarsest level, $u_0 = A_0^{-1}b_0$, has discretization error

$$|||u - u_0||| \le C_1 h_0^2, \tag{3.92}$$

where u is the solution to the continuous problem.

Let $e_j = u - u_j^*$ denote the error on level j, where u_j^* is the iterative approximation on level j. Define the *relative discretization error* [14] by $u_{j-1} - u_j$. This error can be estimated by

$$|||P^{(2)}_{j-1}u_{j-1} - u_j||| \le C_2 h_j^2, \tag{3.93}$$

where $P^{(2)}_{j-1}$ is the second order prolongation from level $j-1$ to level j.

Thus, if the error e_0 on level 0 is interpolated to level 1, we have

$$\begin{aligned}
|||P_0^{(2)}e_0||| &= |||P_0^{(2)}(u-u_0)||| \\
&= |||P_0^{(2)}(u-u_0)-u_1+u_1||| \\
&\leq |||P_0^{(2)}u_0-u_1|||+|||P_0^{(2)}u-u_1||| \qquad (3.94)\\
&\leq C_2h_0^2+C_3|||u-u_1||| \\
&\leq C_2h_0^2+C_1C_3h_1^2.
\end{aligned}$$

If we assume that $\sigma = \frac{h_{j-1}}{h_j}$, then

$$|||P_0^{(2)}e_0||| \leq C_1C_3h_1^2 + C_2(\sigma h_1)^2 = (C_1C_3+\sigma^2C_2)h_1^2. \qquad (3.95)$$

The error of a cascadic multigrid method employing both a second order discretization and interpolation will be bounded by

$$\frac{E_2}{\sigma^{d(l-1)}} \qquad (3.96)$$

on level l.

Standard cascadic multigrid algorithms

Algorithm 3.8 performs the cascadic multigrid method for an $\mathcal{O}(h^4)$ method. On the coarsest level the problem is solved directly using an $\mathcal{O}(h^4)$ discretization (see (2.16) for the two dimensional case).

The solution on the coarsest level is interpolated to the next finer level using a fourth order interpolation, which is similar to doing interpolation plus a smoothing. For example, in two dimensions the solution is known at the points (x_{2i}, y_{2k}), thus these points are injected into the finer grid. For the points $(x_{2i\pm1}, y_{2k})$ and $(x_{2i}, y_{2k\pm1})$, cubic interpolation is used on the injected points. The interpolated value at $(x_{2i\pm1}, y_{2k})$ is as follows:

$$u^j_{m,2k} = -\frac{1}{16}(u^{j-1}_{m-3,2k} - 9u^{j-1}_{m-1,2k} - 9u^{j-1}_{m+1,2k} + u^{j-1}_{m+3,2k}) + \mathcal{O}(h^4), \tag{3.97}$$

where $m = 2i \pm 1$ [14]. An analogous formula holds for the interpolated value at $(x_{2i}, y_{2k\pm1})$.

The points $(x_{2i\pm1}, y_{2k\pm1})$ use the standard five point difference scheme. The calculation of these points is shown in (3.98):

$$u^j_{p,q} = \frac{1}{4}(u^{j-1}_{p-1,q} + u^{j-1}_{p+1,q} + u^{j-1}_{p,q-1} + u^{j-1}_{p,q+1} + h^2 b_{p,q}) + \mathcal{O}(h^4), \tag{3.98}$$

where $p = 2i \pm 1, q = 2k \pm 1$. This sets up a system of equations that can be solved using the Jacobi or Gauss-Seidel iteration.

Once the interpolation has taken place, the approximation is smoothed using symmetric Gauss-Seidel until the energy norm is less than 1/4 that of the energy norm E_2 computed in Algorithm 3.7. This is repeated on all levels with bound $E_2/(4^{(\text{level-1})})$. The initial guess on the finer grids is the fourth order interpolation of the approximation from the previous grid.

The steps in Algorithm 3.9 are identical to those of Algorithm 3.8, except now

Algorithm 3.8 Cascadic multigrid using $\mathcal{O}(h^4)$ method

1. If $j = 0$
   (a) Solve directly
   (b) Interpolate to $j = 1$ using fourth order interpolation
2. If $j > 0$
   (a) Smooth on u until $E^{(j)} < E_2/4^{(j-1)}$
   (b) Interpolate u to level $j + 1$ using fourth order interpolation

the discretization is $\mathcal{O}(h^2)$ and the interpolation is linear.

Algorithm 3.9 Cascadic multigrid using $\mathcal{O}(h^2)$ method

1. If $j = 0$
   (a) Solve directly
   (b) Interpolate to $j = 1$ using linear interpolation
2. If $j > 0$
   (a) Smooth on u until $E^{(j)} < E_2/4^{(j-1)}$
   (b) Interpolate u to level $j + 1$ using linear interpolation

3.6.2 Cascadic multigrid switching order on level $k + 1$

Algorithm 3.10 gives a cascadic multigrid algorithm using a fourth order discretization and fourth order interpolation until level $k+1$, at which point both the discretization and interpolation switch to second order.

On the coarsest level, a direct solve on the fourth order discretization $A_0 u_0 = b_0$

Algorithm 3.10 Cascadic multigrid switching discretization at level k

1. If $j = 0$
   (a) Solve directly using $\mathcal{O}(h^4)$ method
   (b) Interpolate to $j = 1$
2. Else if $j = 1, \ldots, k$
   (a) Smooth $m_j^{(4)}$ times using $\mathcal{O}(h^4)$ method
   (b) Interpolate to level $j+1$ using fourth order interpolation
3. Else if $j = k+1, \ldots, l-1$
   (a) Smooth $m_j^{(2)}$ times using $\mathcal{O}(h^2)$ method
   (b) Interpolate to level $j+1$ using linear interpolation

is used to compute u_0. The discretization error is given as follows:

$$|||u - u_0||| \leq C_4 h_0^4, \tag{3.99}$$

where u is the solution to the continuous problem.

The relative discretization error of the fourth order approximation can be estimated by

$$|||P_{j-1}^{(4)} u_{j-1} - u_j||| \leq C_5 h_j^4, \tag{3.100}$$

where $P_{j-1}^{(4)}$ is the fourth order prolongation from level $j-1$ to level j.

Using the fourth order prolongation, interpolate the solution on level 0 to level 1:

$$u_1 = P_0^{(4)} u_0. \tag{3.101}$$

If the error e_0 is interpolated to level 1, we have

$$\begin{aligned}
|||P_0^{(4)} e_0||| &= |||P_0^{(4)}(u-u_0)||| \\
&= |||P_0^{(4)}(u-u_0) - u_1 + u_1||| \\
&\leq |||P_0^{(4)} u_0 - u_1||| + |||P_0^{(4)} u - u_1||| \\
&\leq C_5 h_0^4 + C_6 |||u - u_1||| \\
&\leq C_5 h_0^4 + C_4 C_6 h_1^4.
\end{aligned} \tag{3.102}$$

Again assuming that $\sigma = \frac{h_{j-1}}{h_j}$, we have

$$|||P_0^{(4)} e_0||| \leq C_4 C_6 h_1^4 + C_5 (\sigma h_1)^4 = (C_4 C_6 + \sigma^4 C_5) h_1^4. \tag{3.103}$$

Now, on level 1, the discretization is again fourth order. We smooth until

$$|||e_1||| = |||u - u_1^*||| \leq C_4 h_1^4 = \frac{C_4}{\sigma^4} h_0^4. \tag{3.104}$$

For $\sigma = 2$, for example, this means that we want the error on level 1 to be $\frac{1}{16}$ of the error on level 0.

The final approximation on level 1 is interpolated to level 2 using the fourth order prolongation and the interpolated solution is smoothed until the error is $\frac{1}{16}$ of the error on the previous level.

This process continues until level k. On level k, the discretization is still fourth order and the smoothing iterations are performed until the error is $\frac{1}{16}$ of that on level

$k-1$. Thus the error on level k satisfies

$$|||e_k||| \leq C_4 h_k^4 = \frac{C_4}{16} h_{k-1}^4 = \frac{C_4}{2^{4k}} h_0^4. \tag{3.105}$$

The final approximation on level k is interpolated using *second order* interpolation $P_k^{(2)}$ to level $k+1$. The discretization on level $k+1$ is also second order. Thus we have

$$\begin{aligned}
|||P_k^{(2)} e_k||| &= |||P_k^{(2)}(u - u_k)||| \\
&= |||P_k^{(2)}(u - u_k) - u_{k+1} + u_{k+1}||| \\
&\leq |||P_k^{(2)} u_k - u_{k+1}||| + |||P_k^{(2)} u - u_{k+1}||| \\
&\leq C_2 h_k^2 + C_3 |||u - u_{k+1}||| \\
&\leq C_2 h_k^2 + C_1 C_3 h_{k+1}^2 \\
&= C_2 (2h_{k+1})^2 + C_1 C_3 h_{k+1}^2 \\
&= 4C_2 h_{k+1}^2 + C_1 C_3 h_{k+1}^2 \\
&= C_7 h_{k+1}^2.
\end{aligned} \tag{3.106}$$

We want to guarantee that

$$|||e_{k+1}||| \leq C_1 h_{k+1}^2; \tag{3.107}$$

thus we smooth on level $k+1$ until (3.107) is met, starting from an initial error on level $k+1$ of $C_7 h_{k+1}^2$.

We have proven the following theorem:

Theorem 3.6.1 *The error of the cascadic multigrid algorithm given in Algorithm 3.10 can be estimated by*

$$\begin{aligned} |||u_l - u_l^*||| &\leq |||u - u_l||| + |||u - u_l^*||| \\ &\leq C_4 h_l^4 + C_1 h_l^2. \end{aligned} \tag{3.108}$$

3.6.3 Errors

Consider a specific example, namely the Poisson equation (2.13) on a uniform mesh of spacing h_j with $\sigma \approx 2$. The second order discretization of (2.13) is

$$4u_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1} = h^2 b_{i,j}, \tag{3.109}$$

The error in the discretization is derived using the Taylor series. For example, the Taylor series for $u(x \pm h)$ is

$$u(x \pm h) = u(x) \pm u'(x)h + \frac{u''(x)h^2}{2!} \pm \frac{u'''(x)h^3}{3!} + \frac{u^{iv}(x)h^4}{4!} \pm \ldots \tag{3.110}$$

Then $-u_{xx}$ can be written as

$$-u_{xx} = \frac{1}{h^2}[-u(x-h,y) + 2u(x,y) - u(x+h,y)] + \frac{1}{12}u_{xxxx}(\xi)h^2. \tag{3.111}$$

Combining this with

$$-u_{yy} = \frac{1}{h^2}[-u(x,y-h) + 2u(x,y) - u(x,y+h)] + \frac{1}{12}u_{yyyy}(\zeta)h^2 \tag{3.112}$$

gives us (3.109). So the second order discretization of $-u_{xx} - u_{yy}$ has a truncation error of

$$\frac{h^2}{12}(u_{xxxx}(\xi) + u_{yyyy}(\zeta)), \tag{3.113}$$

so the error is $\mathcal{O}(h^2)$. The error is bounded by $\frac{h^2}{6}\max_{\xi,\zeta}\{u_{xxxx}(\xi), u_{yyyy}(\zeta)\}$.

The fourth order discretization in two dimensions is as follows:

$$\begin{aligned} &20u_{i,j} - 4(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}) - \\ &(u_{i-1,j-1} + u_{i-1,j+1} + u_{i+1,j-1} + u_{i+1,j+1}) \\ &= \tfrac{h^2}{2}(8b_{i,j} + b_{i-1,j} + b_{i+1,j} + b_{i,j-1} + b_{i,j+1}) \end{aligned} \tag{3.114}$$

Although both sides of this equation are only $\mathcal{O}(h^2)$, when used to solve problems such as Poisson's equation, the discretization is $\mathcal{O}(h^4)$.

Poisson's equation

The problem to be solved is Poisson's equation with Dirichlet boundary conditions given in Equation (3.20). Quasi-uniform triangulations with meshsize parameter given in (3.24) are assumed.

Let $\bar{V}_j$ represent the finite element space corresponding to V_j (see (3.23)) for higher order basis functions (order s) on the coarser levels and let $P^{h,s}$ represent an interpolator for the family of finite elements based on a subdivision of Ω.

For $-2 \leq s \leq 2$, we introduce H^s-regularity to the problem in order to justify using higher order discretizations on some of the coarser grids:

$$\|u\|_s \leq c\,\|b\|_{s-2}, \quad \forall b \in H^{s-2}(\Omega). \tag{3.115}$$

Theorem 3.6.2 *(Regularity Theorem, see Theorem 7.2 [5]) Let a be an H_0^1-elliptic bilinear form with sufficiently smooth coefficient functions.*

1. *If Ω is convex, then the Dirichlet problem is H^2-regular.*

2. *If Ω has a C^s boundary with $s \geq 2$, then the Dirichlet problem is H^s-regular.*

Let the shape functions have an approximation order s; that is, the interpolation error in the H^m-norm, $0 \leq m \leq s$, can be estimated by

$$\|u - P^{h,s}u\|_m \;\; \leq \;\; c\,h^{s-1}|u|_s. \tag{3.116}$$

Suppose V is a closed subset of a Hilbert space and

$$V_h \subset V. \tag{3.117}$$

In addition, assume

$$P^{h,s}\left(V \cap C^k(\Omega)\right) \subset V_h, \tag{3.118}$$

where k is the highest order of the differentiation in the definition of $P^{h,s}$. Then, by Céa's Theorem (Theorem 3.3.1), we can prove the following theorem:

Theorem 3.6.3 [7] *Suppose that conditions (3.116), (3.117) and (3.118) hold. Then the unique solution, $u_h \in V_h$, to the variational problem*

$$a(u_h, v) = b(v), \quad \forall\, v \in V_h$$

satisfies

$$\|u - u_h\|_m \le c\, h^{s-1}\, |u|_s,$$

for $0 \le m \le s$.

PROOF:

$$\begin{aligned}
\|u - u_h\|_m &\le c \min_{v \in V_h} \|u - v\|_m && \text{by Céa's Theorem [7]} \\
&\le c\, \|u - P^{h,s} u\|_m && P^{h,s} u \in V_h, \text{ thus } \ge \min \\
&\le c\, h^{s-1}\, |u|_s && \text{by approximation order assumption}
\end{aligned}$$

■

Theorem 3.6.3 with $m = 1$ enables us to bound $\|u - u_h\|^2_{L^2}$. To illustrate this, define $e = u - u_h$ and let w be the solution of

$$\left\{\begin{array}{l} -\Delta w = e \text{ in } \Omega, \\ u = 0 \text{ on } \Gamma \subset \partial\Omega, \\ \dfrac{\partial u}{\partial \nu} = 0 \text{ on } \partial\Omega \backslash \Gamma. \end{array}\right. \tag{3.119}$$

The variational formulation of this problem is:

$$\text{Find } w \in V \text{ such that } a(w,v) = (e,v), \quad \forall\, v \in V. \tag{3.120}$$

The solution to (3.120) exists uniquely, since $u - u_h \in V'$, the dual of V. Therefore,

$$\begin{array}{rcll} \|u-u_h\|_0^2 & = & (u-u_h, u-u_h) & \text{by the definition of norm} \\ & = & (e, u-u_h) & \text{by the definition of } e \\ & = & a(w, u-u_h) & \text{by the variational formulation} \\ & = & a(u-u_h, w-P^{h,2}w) & \begin{array}{rcl} a(u-u_h, P^{h,2}w) & = & a(e, P^{h,2}w) \\ & = & 0 \,\text{orthogonal} \end{array} \\ & \leq & c\,\|u-u_h\|_1 \|w-P^{h,2}w\|_1 & \text{by Schwarz' inequality} \\ & \leq & c\,h\,\|u-u_h\|_1 |w|_2. & \text{by Theorem 3.6.3} \end{array} \tag{3.121}$$

Suppose that the equivalent problems (3.119) and (3.120) have the property that

$$|w|_2 \leq c\,\|e\|_0. \tag{3.122}$$

Thus we have the well known theorem (cf. [7]):

Theorem 3.6.4 *Suppose that conditions (3.116), (3.117), (3.118), and (3.122) hold. Then*

$$\begin{aligned} \|u - u_h\|_0 &\leq c\,h\,\|u - u_h\|_1 \\ &\leq c\,h^s\,|u|_s. \end{aligned}$$

PROOF:

$$\begin{aligned} \|u - u_h\|_0^2 &\leq c\,h\,\|u - u_h\|_1 |w|_2 && \text{from (3.121)} \\ &\leq c\,h\,\|u - u_h\|_1 \|u - u_h\|_0 && \text{from (3.122)} \end{aligned}$$

Dividing both sides by $\|u - u_h\|_0$, we get

$$\begin{aligned} \|u - u_h\|_0 &\leq c\,h\,\|u - u_h\|_1 \\ &\leq c\,h\,h^{s-1}\,|u|_s && \text{by Theorem 3.6.3} \\ &= c\,h^s\,|u|_s \end{aligned}$$

■

By Theorem 3.6.3, we have

$$\begin{aligned}
|||u-u_j||| &\leq c\,\|u-u_j\|_1 && \text{by (3.31)} \\
&\leq c\,h^{s-1}\,|u|_s && \text{by Theorem 3.6.3} \\
&\leq c\,h^{s-1}\,\|u\|_s && \text{norm comparison} \\
&\leq c\,h^{s-1}\,\|b\|_{s-2} && \text{by (3.115)}
\end{aligned} \tag{3.123}$$

Thus, the approximation error of the finite element method is given in energy norm as

$$|||u-u_j||| \leq c\,h_j^{s-1}\,\|b\|_{s-2}, \quad j=0,\ldots,l. \tag{3.124}$$

On the coarser grids where the higher order discretizations are used, we have

Lemma 3.6.5 *The smoother on the coarser levels fulfills*

$$|||\mathfrak{S}^{SGS}_{j,m_j} v_j||| \;\leq\; c\,\frac{h_j^{s-1}}{m_j^{(1+s)\gamma}}\|v_j\|_s \tag{3.125}$$

Proof:

For $t \geq s$, $\hat{c}^{-t/2}|||v|||_t \geq \hat{c}^{-s/2}|||v|||_s$ [5], where $\hat{c}$ is the constant of ellipticity. Let $c_s(v_j) = \frac{m_j^{s/2}}{h_j^s}$ for $1 \leq s \leq 2$.

For $0 < s \leq 2$,

$$
\begin{aligned}
|||v_j|||_s^2 &= \sum_{i=1}^{n_j} c_i^2 (\lambda_i^{(j)})^s \\
&\leq \sum_{i=1}^{n_j} c_i^2 \lambda_{n_j}^{(j)} (\lambda_i^{(j)})^{s-1} \\
&= \lambda_{n_j}^{(j)} \sum_{i=1}^{n_j} c_i^2 (\lambda_i^{(j)})^{s-1} \\
&= \lambda_1^{(j)} |||v|||_{s-1}^2 \\
&\leq c|||v|||_{s-1}^2 \\
&\leq c\|v_j\|_{s-1}^2 \\
&\leq c\|v_j\|_s^2 .
\end{aligned}
\tag{3.126}
$$

Thus $|||v_j|||_s \leq c\|v\|_s$ and

$$
\begin{aligned}
|||\mathfrak{S}_{j,m_j}^{SGS} v_j||| &\leq c \frac{h_j^{-1}}{m_j^{1/2}} \|v_j\|_0 \\
&\leq c \frac{h_j^{-1}}{m_j^{1/2}} |||v_j|||_0 \\
&\leq c \frac{h_j^{-1}}{m_j^{1/2}} |||v_j|||_s \\
&\leq c \frac{h_j^{-1}}{m_j^{1/2}} \|v_j\|_s \\
&\leq c\, c_s(v_j) \tfrac{h_j^{s-1}}{m_j^{(1+s)/2}} \|v_j\|_s .
\end{aligned}
\tag{3.127}
$$

■

3.7 Parabolic problems

The parabolic problem [23] to be solved is as follows: Find $u(x,t)$ such that

$$\begin{cases} \frac{\partial u}{\partial t} + \mathcal{L}u &= b \qquad \text{in } \Omega \times [0, \mathbf{t}], \\ u(x,t) &= 0 \qquad \text{in } \partial\Omega \times [0, \mathbf{t}], \\ u(x,0) &= u_0(x), \end{cases} \tag{3.128}$$

where $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$, is a bounded domain, $b \in L^2(\Omega)$, and $\mathcal{L}$ is the elliptic operator

$$\mathcal{L}u = -\sum_{i,j=1}^{d} \frac{\partial}{\partial x_i}(a_{ij}(x)\frac{\partial u}{\partial x_j}). \tag{3.129}$$

For $c, C > 0$, we assume that

$$c\phi^t\phi \leq \sum_{i,j=1}^{d} a_{ij}(x)\phi_i, \phi_j \leq C\phi^t\phi, \tag{3.130}$$

for all $x \in \Omega, \phi \in \mathbb{R}^d$. This guarantees ellipticity of the operator $\mathcal{L}$, and illustrates equivalence of the energy and L^2 norms.

The variational form of (3.128) is as follows: Find $u \in H_0^1(\Omega)$ with $u(x,0) = u_0(x)$ such that

$$(\frac{\partial u}{\partial t}, v) + B(u,v) = (b,v), \tag{3.131}$$

for all $v \in H_0^1(\Omega), t \in [0, \mathbf{t}]$, and

$$B(u,v) = \int_\Omega \sum_{i,j=1}^{d} a_{ij} \frac{\partial u}{\partial x_j} \frac{\partial v}{\partial x_i} dx, \tag{3.132}$$

for all $u, v \in H^1(\Omega)$ and

$$(b, v) = \int_\Omega bvdx. \tag{3.133}$$

The backward Euler and Crank-Nicolson schemes are used for the time discretization. The backward Euler method approximates $\frac{\partial u}{\partial t}$ at time step n using a divided difference with the unknown u^n on the right hand side:

$$\frac{u^n - u^{n-1}}{\Delta t_n} = b - \mathcal{L}u^n. \tag{3.134}$$

The Crank-Nicolson scheme averages the backward Euler and the forward Euler ((3.134) with u^{n-1} on the right hand side):

$$\frac{u^n - u^{n-1}}{\Delta t_n} = \frac{1}{2}[(b - \mathcal{L}u^n) + (b - \mathcal{L}u^{n-1})]. \tag{3.135}$$

These yield the following problem: Given $g_{n-1} \in H^{-1}(\Omega)$, find $w \in H_0^1(\Omega)$ such that

$$A_\tau(w, v) = \tau^{-1}(w, v) + B(w, v) = (g_{n-1}, v), \tag{3.136}$$

for all $v \in H_0^1(\Omega)$, where τ is the time step parameter. Using the backward Euler scheme with $w = u^n - u^{n-1}$ and $\tau = \Delta t_n$, we get

$$(g_{n-1}, v) = (b, v) - B(u^{n-1}, v). \tag{3.137}$$

The Crank-Nicolson scheme with $w = u^n - u^{n-1}$ and $\tau = \Delta t_n/2$ gives us

$$(g_{n-1}, v) = 2((b, v) - B(u^{n-1}, v)). \tag{3.138}$$

For the backward Euler we have

$$\frac{u^n - u^{n-1}}{\Delta t_n} = b - \mathcal{L}u^n, \tag{3.139}$$

thus

$$\frac{1}{\Delta t_n}(u^n - u^{n-1}, v) = (b, v) - B(u^n, v). \tag{3.140}$$

This gives us

$$\begin{aligned}
(g_{n-1}, v) &= \tau^{-1}(w, v) + B(w, v) \\
&= \tfrac{1}{\Delta t_n}(u^n - u^{n-1}, v) + B(u^n - u^{n-1}, v) \\
&= (b, v) - B(u^n, v) + B(u^n, v) - B(u^{n-1}, v) \\
&= (b, v) - B(u^{n-1}, v).
\end{aligned} \tag{3.141}$$

For the Crank-Nicolson case we have

$$\frac{u^n - u^{n-1}}{\Delta t_n} = \frac{1}{2}[(b - \mathcal{L}u^n) + (b - \mathcal{L}u^{n-1})] - \mathcal{L}u^n, \tag{3.142}$$

thus

$$\frac{1}{\Delta t_n}(u^n - u^{n-1}, v) = (b, v) - \frac{1}{2}B(u^n, v) - \frac{1}{2}B(u^{n-1}, v). \tag{3.143}$$

Then,

$$
\begin{aligned}
(g_{n-1}, v) &= \tau^{-1}(w, v) + B(w, v) \\
&= \tfrac{2}{\Delta t_n}(u^n - u^{n-1}, v) + B(u^n - u^{n-1}, v) \\
&= 2(b, v) - B(u^n, v) - B(u^{n-1}, v) + B(u^n, v) - B(u^{n-1}, v) \\
&= 2((b, v) - B(u^{n-1}, v)).
\end{aligned}
\tag{3.144}
$$

The partition of Ω on level j is a quasi-uniform triangulation T_j with mesh size $h_j = h_0 2^{-j}$. T_{j+1} is formed by joining the midpoints of each triangle on T_j. V_j is the P1 conforming finite element space on T_j.

The Richardson iteration is often used as the smoother in the parabolic case as it guarantees convergence and is easier to analyze. In Algorithm 3.11, $M_{j,\tau}$ represents the Richardson iteration procedure, where

$$
\begin{aligned}
u_j - M_{j,\tau}^{m_j} u_j^0 &= S_{j,\tau}^{m_j}(u_l - u_l^0) \\
&= (I - R_{j,\tau} A_{j,\tau})(u_j - u_j^0).
\end{aligned}
\tag{3.145}
$$

In the procedure, $R_{j,\tau} = (\lambda_j + \tau^{-1})^{-1} I$ where $\lambda_j = \mathcal{O}(h_j^{-2})$ and $A_{j,\tau} : V_j \to V_j$ is defined as $(A_{j,\tau} u_j, v_j) = A_\tau(u_j, v_j)$ for all $u_j, v_j \in V_j$.

Algorithm 3.11 Cascadic multigrid using Richardson iteration

1. Set $u_0^0 = u_0^* = u_0$
2. For $j = 1, ..., l$
   (a) $u_j^0 = u_{j-1}^*$
   (b) $u_j^{m_j} = M_{j,\tau}^{m_j} u_j^0$
   (c) Set $u_j^* = u_j^{m_j}$

Define the τ-norm as

$$\|v\|_\tau^2 = \tau^{-1}(v,v) + B(v,v) \tag{3.146}$$

for all $v \in H_0^1(\Omega)$.

Algorithm 3.11 will be *optimal* on level j if both *accuracy in the* τ*- norm*

$$\|u_j - u_j^*\|_\tau \approx \|u - u_j\|_\tau$$

(the iterative error is comparable to the approximation error) and *multigrid complexity*

$$\text{amount of work } = \mathcal{O}(n_j),$$

where n_j=dimV_j, are obtainable simultaneously.

Lemma 3.7.1 [23] *For* $S_{j,\tau}^{m_j}$ *and any* $v \in V_j$,

$$\begin{aligned} (i) \quad \|S_{j,\tau}^{m_j} v\|_\tau &\le (1+\tau^{-1}\lambda_j^{-1})^{-m_j}\|v\|_\tau, \\ (ii) \quad \|S_{j,\tau}^{m_j} v\|_\tau &\le c\frac{h_j^{-1}}{m_j^{\frac{1}{2}}}(1+\tau^{-1}\lambda_j^{-1})^{\frac{-m_j}{2}}\|v\|_0, \ \textit{for } m_j \ge 2. \end{aligned} \tag{3.147}$$

Lemma 3.7.2 proves the convergence of the cascadic algorithm:

Lemma 3.7.2 [23] *The error of the cascadic multigrid method for the parabolic problem with the Richardson iteration (Algorithm 3.11) can be estimated by*

$$\|u_j - u_j^*\|_\tau \le c\sum_{j=0}^{l}\frac{h_j}{m_j^{\frac{1}{2}}}\prod_{i=0}^{l-j-1}(1+\tau^{-1}\lambda_{l-i}^{-1})^{-m_{l-i}}(1+\tau^{-1}\lambda_j^{-1})^{-\frac{m_j}{2}+1}\|g\|_0, \tag{3.148}$$

where $\prod_{i=0}^{-1} = 0$.

Since $h_j \le 2^{l-j} h_l$, we consider sequences $m_1, \ldots, m_l$ of the kind

$$m_j = \lceil \beta^{l-j} m_l \rceil \tag{3.149}$$

for some fixed $\beta > 1$. If $\tau \le \lambda_l^{-1}$, then the Richardson iteration guarantees good convergence [23]; thus we will focus on $\tau \ge \lambda_l^{-1}$. For fixed τ, there exists $\gamma_0 \in (0,1)$ such that $\tau \le \frac{\lambda_l^{-1}}{\gamma_0}$.

Lemma 3.7.3 [23] *If* $\tau \le \frac{\lambda_l^{-1}}{\gamma_0}$, *then*

$$\|u_l - u_l^*\|_\tau \le c \frac{h_l}{m_l^{\frac{1}{2}}} \sum_{j=0}^{l} \left(\frac{2}{\beta^{\frac{1}{2}}(1+\gamma_0)} \right)^j \|g\|_0, \tag{3.150}$$

for $m_l \ge 2$.

Theorem 3.7.4 [23] *For* $\beta > 0$ *such that* $m_j = \lceil \beta^{l-j} m_l \rceil$ *and* $\tau \le \frac{\lambda_l^{-1}}{\gamma_0}$, *the accuracy of the parabolic cascadic multigrid with the Richardson iteration is*

$$\|u_l - u_l^*\|_\tau \le c \frac{h_l}{m_l^{\frac{1}{2}}} \frac{1}{1 - \frac{2}{\beta^{\frac{1}{2}}(1+\gamma_0)}} \|g\|_0, \tag{3.151}$$

for $\beta > \left(\frac{2}{1+\gamma_0}\right)^2$.

Lemma 3.7.5 [23] *For $\beta < 2^d$, where d is the dimension of Ω, the computational cost of the cascadic multigrid is proportional to*

$$\sum_{j=1}^{l} m_j n_j \leq c \frac{1}{1-\frac{\beta}{2^d}} m_l n_l. \tag{3.152}$$

Thus for large β we can obtain optimal accuracy and for small β we can obtain optimal complexity. To obtain both, β must be bounded as follows: $\left(\frac{2}{1+\gamma_0}\right)^2 < \beta < 2^d$. For $d = 2, 3$, this will hold for any $\gamma_0 \in (0, 1)$. In the one dimensional case ($d = 1$), γ_0 must be larger than $\sqrt{2} - 1$, and hence τ must be less than $\lambda_l^{-1}(\sqrt{2} + 1)$, thus the time step cannot be too large.

Note: The cascadic multigrid with the Richardson iteration has optimal accuracy and complexity for 3D elliptic problem, but only nearly optimal complexity for 2D. In 1D the convergence is very slow.

The Richardson procedure $M_{j,\tau}^{m_j}$ in Algorithm 3.11 can be replaced by m_j steps of the conjugate gradient iteration.

The error of the conjugate gradient method on level j is

$$\|u_j - M_{j,\tau}^{m_j} u_j^0\|_\tau = \min_{p \in P_{m_j}, p(0)=1} \|p(A_{j,\tau})(u_j - u_j^0)\|_\tau, \tag{3.153}$$

where u_j^0 is the initial value on level j and P_{m_j} is the set of polynomials p with degree less than or equal to m_j.

Lemma 3.7.6 [23] *The accuracy of the cascadic multigrid with the conjugate gradient iteration is*

$$\|u_l - u_l^*\|_\tau \leq \begin{cases} c\frac{1}{1-\left(\frac{2}{\beta}\right)}\frac{h_l}{m_l}\|g\|_0 & \text{for } \beta > 2 \\ cl\frac{h_l}{m_l}\|g\|_0 & \text{for } \beta = 2. \end{cases} \tag{3.154}$$

The two and three dimensional parabolic problems are accurate with optimal complexity for the conjugate gradient method. The one dimensional problem is nearly optimal in this case (see Theorem 3.7.7).

Theorem 3.7.7 [23] *For one dimensional parabolic problems, if* $\beta = 2$ *and* $m_l = \lceil m_* l \rceil$*, then the error of the cascadic multigrid with the conjugate gradient iteration is*

$$\|u_j - u_j^*\|_\tau \leq c\frac{h_l}{m_*}\|g\|_0, \tag{3.155}$$

and the complexity is given by

$$\sum_{j=1}^{l} m_j n_j \leq c m_* n_l (1 + \log n_l)^2. \tag{3.156}$$

3.8 Summary

In the early stages of cascadic multigrid, the problem was over solved on coarser levels using a lower order method. Since the problem can only be solved up to the truncation error on these levels, the approximation theory does not make sense. A higher order method would yield a better approximation on each level. However, an algorithm using a higher order method on every level is not computationally sound.

The variable order cascadic multigrid algorithm combines the benefits of a higher order algorithm on the coarser levels with the utility of a lower order algorithm on the finer levels.

Chapter 4

Numerical Experiments

4.1 One Dimensional Problem

The problem to be solved is Poisson's equation with Dirichlet boundary conditions:

$$\begin{gathered} -\tfrac{d^2u}{dx^2} = b(x) \\ u(0) = u(1) = 0 \end{gathered} \tag{4.1}$$

The linear interpolation used in the $\mathcal{O}(h^2)$ algorithm is the standard linear interpolation:

$$\begin{aligned} u_{2i}^{(j)} &= u_i^{(j-1)} \\ u_{2i+1}^{(j)} &= \tfrac{1}{2}(u_i^{(j-1)} + u_{i+1}^{(j-1)}) \end{aligned}$$

The fourth order interpolation comes from Hyman's Local Inversion Method [17]. First the coarse points are injected and then the unknown fine points are solved for locally as with a smoother:

$$
\begin{aligned}
u_{2i}^{(j)} &= u_i^{(j-1)} \\
u_{2i+1}^{(j)} &= \tfrac{1}{2}\left(u_i^{(j-1)} + u_{i+1}^{(j-1)} + h^2 b_{2i+1}^{(j)}\right)
\end{aligned}
\tag{4.2}
$$

The projection used in the V cycle is the weighted restriction:

$$u_i^{(j-1)} = \frac{1}{4}(u_{2i-1}^{(j)} + 2u_{2i}^{(j)} + u_{2i+1}^{(j)}).$$

The second order discretization involves the standard three point formula:

$$-u_{i-1} + 2u_i - u_{i+1} = h^2 b_i$$

The fourth order discretization has the following scheme:

$$-u_{i-1} + 2u_i - u_{i+1} = h^2[b_{i-1} + b_{i+1} + 10b_i]$$

The smoother used is symmetric Gauss-Seidel.

4.1.1 Sample Problem

The sample problem has solution $u = x\sin(\pi x)$, shown in Figure 4.1.

Algorithms 3.8, 3.9 and 3.10 were used to solve this problem. The iterations for each method are shown in Figure 4.2, where the initial number of interior grid points is 63 and the number of levels is seven.

Figure 4.2 shows that the second order method requires several more iterations per level than does the fourth order method. In fact, the fourth order discretization

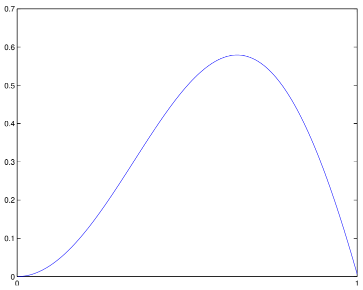

Figure 4.1: Solution for One Dimensional Sample Problem

and interpolation yield an error so small so that only one iteration of the smoother is required throughout the algorithm. The variable order algorithm is identical to the fourth order algorithm until the fourth level, thus requires the same number of iterations. After the switch to second order, the number of iterations of the variable order algorithm is the same as the second order algorithm.

The error in the second order discretization (by the Taylor polynomial), is

$$\frac{h^2}{12} \max_{\xi} |u''''(\xi)| \approx 12.34h^2.$$

Component-wise, the largest error in the discretization is 1.3789×10^{-4}. The energy norm of the discretization error is $|||u - u_0||| = 5.6757 \times 10^{-6} = \mathcal{O}(h^3)$.

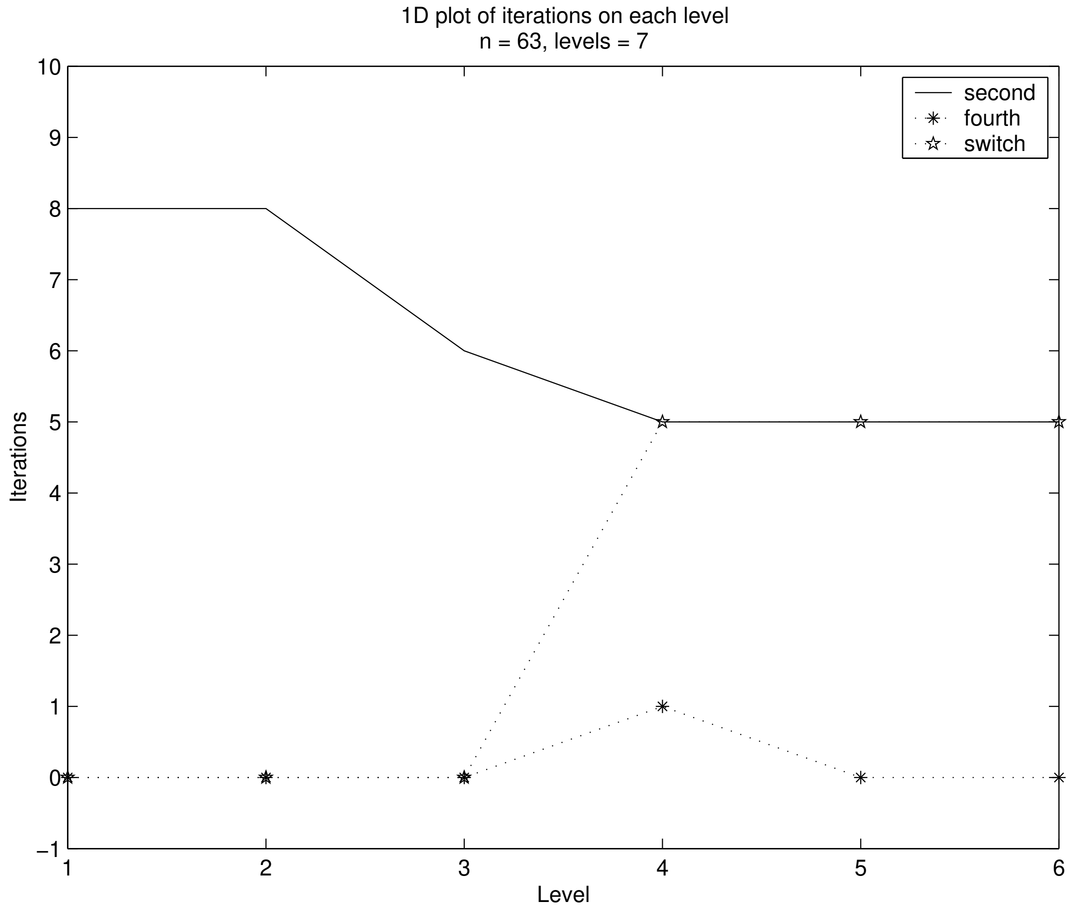

Figure 4.2: Iterations for One Dimensional Sample Problem

4.2 Two Dimensional Problem

4.2.1 Sample Problem 1

The problem to be solved is Poisson's equation with Dirichlet boundary conditions:

$$\begin{aligned} -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} &= b(x,y) \quad \text{in } D^2 \\ u(x,y) &= 0 \qquad\quad \text{on } \partial D^2 \end{aligned} \tag{4.3}$$

The solution to this problem is $e^{xy}\sin(\pi x)\sin(\pi y)$, shown in Figure 4.3.

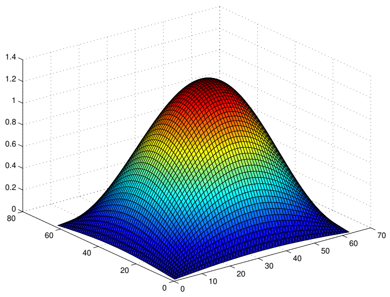

Figure 4.3: Solution for Two Dimensional Sample Problem 1

The interpolation used in the $\mathcal{O}(h^2)$ algorithm is the standard bilinear interpolation:

$$\begin{aligned}
u^{(l)}_{2i,2j} &= u^{(l-1)}_{i,j}\\
u^{(l)}_{2i+1,2j} &= \tfrac{1}{2}(u^{(l-1)}_{i,j} + u^{(l-1)}_{i+1,j})\\
u^{(l)}_{2i,2j+1} &= \tfrac{1}{2}(u^{(l-1)}_{i,j} + u^{(l-1)}_{i,j+1})\\
u^{(l)}_{2i+1,2j+1} &= \tfrac{1}{4}(u^{(l-1)}_{i,j} + u^{(l-1)}_{i+1,j} + u^{(l-1)}_{i,j+1} + u^{(l-1)}_{i+1,j+1})
\end{aligned}$$

The fourth order interpolation used is James Hyman's Local Inversion Method [17]. First the coarse points are injected and then the unknown fine points are solved for locally as with a smoother.

The solution is known at the points (x_{2i}, y_{2j}); thus these points are injected into the finer grid. For the points (x_{2i+1}, y_{2j+1}), the standard five point difference approximation is rotated by 90°. These points are called $\times$ points because they use the surrounding corner points to approximate the central point. They are calculated as follows:

$$u_{2i+1,2j+1} = \frac{1}{4}(u_{2i,2j} + u_{2i+2,2j} + u_{2i,2j+2} + u_{2i+2,2j+2} + 2h^2 b_{2i+1,2j+1}) + \mathcal{O}(h^4).$$

The points (x_{2i}, y_{2j+1}) and (x_{2i+1}, y_{2j}) use the standard five point difference scheme, called the + scheme. The calculation of these points involves the points at which the solution is known and the $\times$ points:

$$u_{k,l} = \frac{1}{4}(u_{k-1,l} + u_{k+1,l} + u_{k,l-1} + u_{k,l+1} + h^2 b_{k,l}) + \mathcal{O}(h^4),$$

where $k = 2i+1, l = 2j$ or $k = 2i, l = 2j+1$.

The projection used in the V cycle is the full weighted restriction:

$$\begin{aligned} u_{i,j}^{(j-1)} &= \tfrac{1}{16}[u_{2i-1,2j-1}^{(j)} + u_{2i-1,2j+1}^{(j)} + u_{2i+1,2j-1}^{(j)} + u_{2i+1,2j+1}^{(j)} \\ &\quad +2(u_{2i,2j-1}^{(j)} + u_{2i,2j+1}^{(j)} + u_{2i-1,2j}^{(j)} + u_{2i+1,2j}^{(j)}) + 4u_{2i,2j}^{(j)}]. \end{aligned}$$

The second order discretization involves the standard five point formula given in Equation (2.14).

The fourth order discretization uses the nine point formula given in Equation (2.16).

$$20u_{i,j} - 4(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}) - (u_{i-1,j-1} + u_{i-1,j+1} + u_{i+1,j-1} + u_{i+1,j+1})$$

$$= \frac{h^2}{2}(8b_{i,j} + b_{i-1,j} + b_{i+1,j} + b_{i,j-1} + b_{i,j+1}).$$

Figure 4.4 is a comparison of the three cascadic Multigrid algorithms given in Algorithms 3.9, 3.8, and 3.10, with grid size $n = 7$ in both directions. The first line represents Algorithm 3.9 which uses a second order discretization and bilinear interpolation. The next line represents Algorithm 3.8 which uses a fourth order discretization and fourth order interpolation, and the third represents Algorithm 3.10 which switches from a fourth order method to a second order method on the finest level.

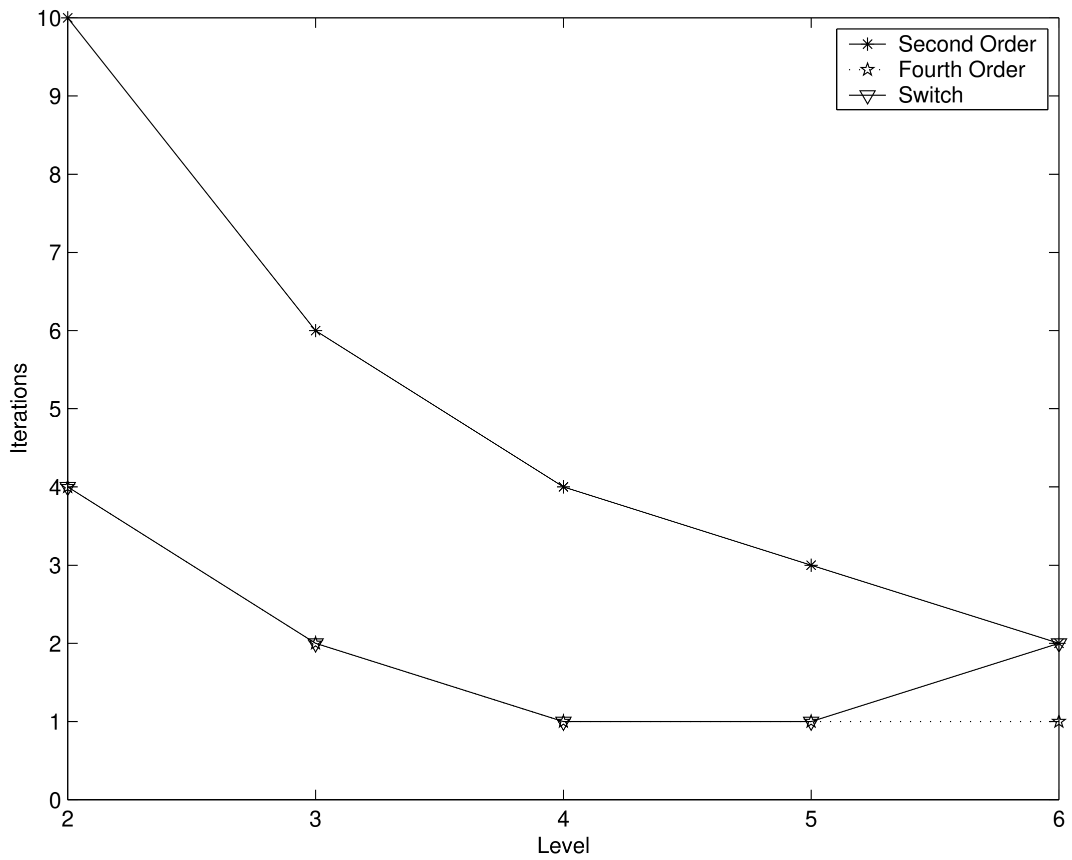

Figure 4.4: Iterations for Two Dimensional Sample Problem 1

As in the one dimensional case, the second order algorithm requires more iterations than the fourth order algorithm, as shown in Figure 4.4. The variable order algorithm has the same number of iterations as the fourth order algorithm until it switches to second order on the finest level.

4.2.2 Sample Problem 2

The problem to be solved represents the flow through a field Ω as follows:

$$\begin{aligned}
\text{div } u &= f && \text{in } \Omega = (0,1) \times (0,1) \\
u &= -K\nabla p && \text{in } \Omega = (0,1) \times (0,1) \\
p &= 100 && \text{on } \Gamma_1 \\
u \cdot n &= 0 && \text{on } \Gamma_2 \\
u \cdot n &= 10 && \text{on } \Gamma_3
\end{aligned} \tag{4.4}$$

where

$$K = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} \tag{4.5}$$

and Γ_1, Γ_2 and Γ_3 are indicated in Figure 4.5 [18]. The black square in the center of the domain indicates the region over which K may be varied.

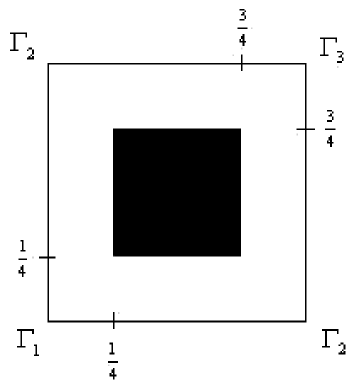

Figure 4.5: Domain Ω for Two Dimensional Sample Problem 2

In this problem, p represents pressure and u represents Darcy velocity. The domain in Figure 4.5 is divided into square cells each of side length $h = 1/n$ where $n = 4 \cdot 2^l$ with level $l = 0, 1, 2, \ldots$. The set of cells is denoted by $\mathcal{T}_h$ and the set of edges by $\mathcal{A}_h$. The discrete problem corresponding to the problem defined in Equation (4.5) is: Find $\{p_T\}_{T \in \mathcal{T}_h}$ and $\{u_E\}_{E \in \mathcal{A}_h}$ such that

$$\begin{aligned}
\int_\Omega K^{-1} \sum_{F \in \mathcal{A}_h} u_F \vec{v}_F \cdot \vec{v}_E - \int_\Omega \sum_{T \in \mathcal{T}_h} p_T \chi_T \ \mathrm{div}\ \vec{v}_E &= \int_{\Omega_D} -\bar{p} \vec{v}_h \cdot \vec{n}, \quad E \in \mathcal{A}_h, E \not\subseteq \Gamma_D \\
\int_\Omega \mathrm{div} \sum_{E \in \mathcal{A}_h} u_E \vec{v}_E \chi_T &= \int_\Omega f \chi_T, \qquad T \in \mathcal{T}_h \\
u_E &= g|E|, \qquad E \subset \Gamma_N,
\end{aligned} \tag{4.6}$$

where $\Gamma_D = \Gamma_1$ is the Dirichlet boundary and Γ_N is the Neumann boundary (Γ_2 and Γ_3).

The lower order discretization uses linear finite elements over each square and the higher order discretization uses quadratic finite elements.

In Figure 4.6, the first line represents the cascadic algorithm using linear finite elements, the second line represents the algorithm with quadratic finite elements, and the third represents a switch from the linear finite elements on the coarser levels to quadratic finite elements on the finest level. The lower order algorithm requires more iterations than does the higher order algorithm. The variable order algorithm has the same number of iterations as the higher order algorithm until it switches on the finest level, where it requires fewer iterations than the lower order algorithm.

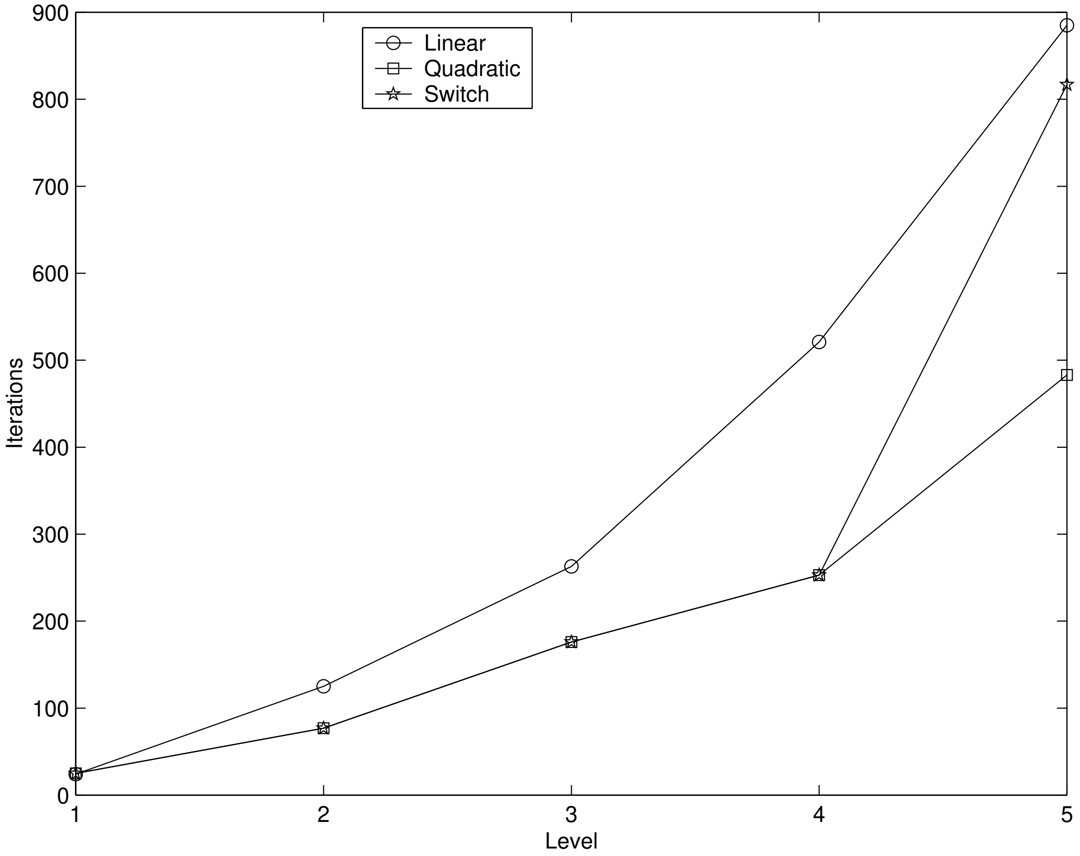

Figure 4.6: Iterations for Two Dimensional Sample Problem 2

4.3 Three Dimensional Problem

The problem to be solved is Poisson's equation with Dirichlet boundary conditions:

$$\begin{aligned} -\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} - \frac{\partial^2 u}{\partial z^2} &= b(x,y,z) \quad \text{in } D^3 \\ u(x,y,z) &= 0 \qquad\qquad\;\; \text{on } \partial D^3 \end{aligned} \tag{4.7}$$

The interpolation used in the $\mathcal{O}(h^2)$ algorithm is the standard trilinear interpolation [26], in which the coarse grid points are injected into the fine grid and the remaining points are determined by averaging the neighboring two, four or eight points, depending on the location.

The fourth order interpolation comes from James Hyman's Local Inversion Method [17], in which the coarse points are injected and the unknown fine points are solved for locally as with a smoother.

The solution is known at the points (x_{2i}, y_{2j}), thus these points are injected into the finer grid. The fine grid points which are located at the center of a cube of known points are determined by averaging the points of the cube. The remaining points are computed using a weighted average of the surrounding octahedral points.

The projection used in the V cycle is the full weighted restriction.

The second order discretization involves the standard seven point formula [26]:

$$6u_{i,j,k} - (u_{i-1,j,k} + u_{i+1,j,k} + u_{i,j-1,k} + u_{i,j+1,k} + u_{i,j,k-1} + u_{i,j,k+1}) = h^2 b_{i,j,k}$$

For the fourth order discretization, the 19 point compact approximation scheme is used [26]:

$$\begin{aligned} 24u_{i,j,k} &- 2(u_{i+1,j,k} + u_{i-1,j,k} + u_{i,j+1,k} + u_{i,j-1,k} + u_{i,j,k+1} + u_{i,j,k-1}) \\ &- (u_{i+1,j+1,k} + u_{i-1,j+1,k} + u_{i+1,j-1,k} + u_{i-1,j-1,k} + u_{i+1,j,k+1} \\ &+ u_{i-1,j,k+1} + u_{i,j+1,k+1} + u_{i,j-1,k+1} + u_{i+1,j,k-1} + u_{i-1,j,k-1} \\ &+ u_{i,j+1,k-1} + u_{i,j-1,k-1}) \\ &= \tfrac{h^2}{2}(6b_{i,j,k} + b_{i+1,j,k} + b_{i-1,j,k} + b_{i,j+1,k} + b_{i,j-1,k} + b_{i,j,k+1} + b_{i,j,k-1}). \end{aligned} \tag{4.8}$$

Figure 4.7 gives a comparison of the three cascadic multigrid algorithms given in Algorithms 3.9, 3.8, and 3.10. The first line represents Algorithm 3.9 which uses a second order discretization and bilinear interpolation. The next line represents Algorithm 3.8 which uses a fourth order discretization and fourth order interpolation, and the third represents Algorithm 3.10 which switches from a fourth order method to a second order method on the finest level.

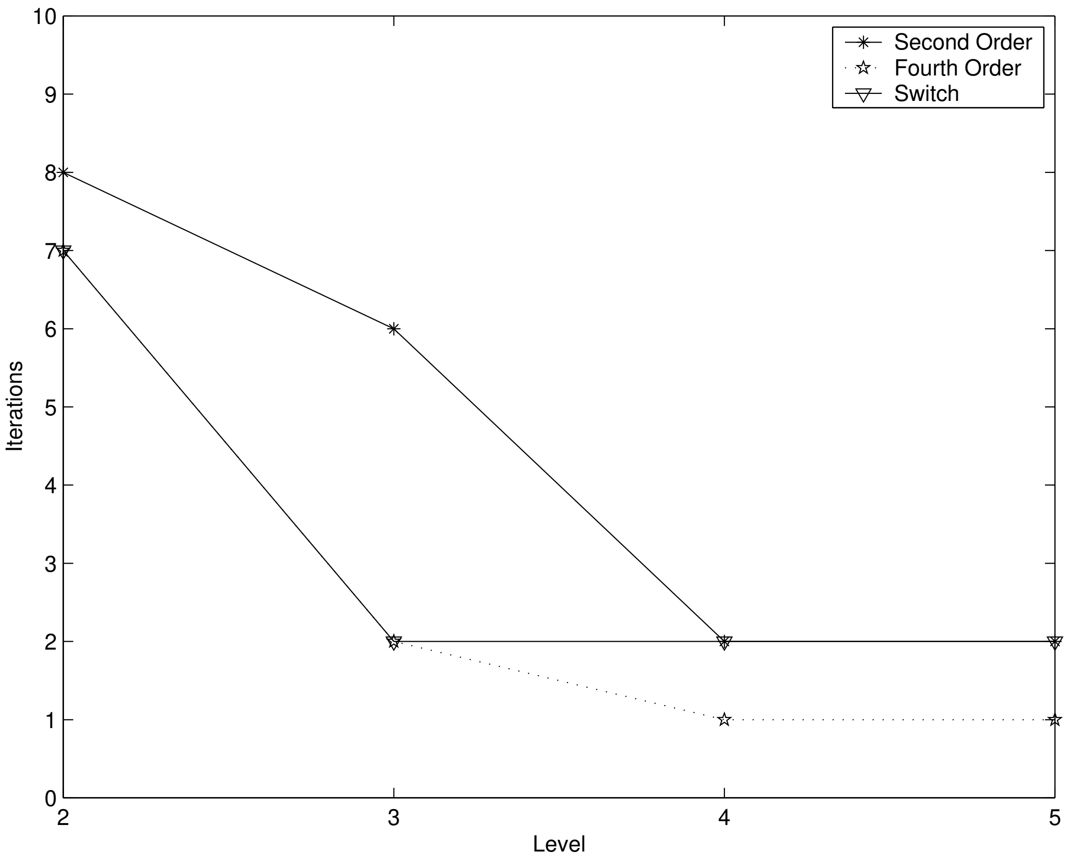

Figure 4.7: Iterations for Three Dimensional Sample Problem

Figure 4.7 shows the iterations for each of the three algorithms. The second order algorithm requires the most iterations, thus is slowest to converge. The fourth order algorithm requires the fewest iterations, and the iterations for the variable order algorithm follow the fourth order iterations until the switch occurs.

Chapter 5

Conclusion

The cascadic preconditioned conjugate gradient methods for elliptic boundary value problems both simplify the original cascadic method of Deuflhard, Leinen and Yserentant and make it more robust. The cascadic conjugate gradient method without preconditioning is shown to be accurate with optimal complexity for elliptic problems with H^2-regularity in two dimensions with quasi-uniform triangulations.

The cascadic multigrid method using the P1 conforming element for the three dimensional Poisson's problem with Dirichlet boundary conditions is accurate with optimal complexity for any choice of energy reducing smoother (e.g., symmetric Gauss-Seidel, SSOR, damped Jacobi, or conjugate gradients). The two dimensional problem has optimal complexity using conjugate gradients, and is nearly optimal for the standard iterative smoothers.

This multigrid optimality for finite differences and finite elements is with respect to the energy norm. Cascadic multigrid cannot be optimal with respect to the L^2 norm for linear finite elements. Higher order finite elements, on the other hand, do allow optimality.

Optimality is also obtained for saddle point problems. The technique used for saddle point problems is also useful for nonconforming finite elements.

The previous results for the second order elliptic problem are valid for other conforming and nonconforming elements. For the plate bending problem, conjugate gradients give only nearly optimal complexity, and other smoothers or roughers cannot be used at all.

For parabolic problems, multigrid is usually used only for large time steps. For small time steps (i.e., $t \leq Ch^2$, where h is the mesh size), a standard iterative method guarantees good convergence for the discrete system. Two and three dimensional parabolic problems are accurate with optimal complexity for conjugate gradients and traditional iterative methods. The one dimensional parabolic problem is nearly optimal for conjugate gradients and is optimal for traditional methods when the time step is restricted to not be too large.

In the variable order cascadic multigrid algorithm, a higher order discretization is used on the coarser levels where the solution is not as expensive due to a smaller number of grid points. A higher order interpolation is also used on these coarser levels. The higher order method allows us to over solve the problem, which is not possible using a lower order method. On the finer levels, both the interpolation and discretization are changed to a lower order method in order to lower computational costs, match the desired order of the solution, or to avoid problems with continuity constraints. These new results show that this method accelerates the convergence of a standard cascadic multigrid algorithm.

Chapter 6

Glossary

$a(\cdot,\cdot) : H_0^1 H^1 \to H^1$	H_0^1-elliptic bilinear form
$\|u\|_s^2 = \|u\|_{H^s}^2 = (u,u)_{H^s} = \sum_{\|\alpha\|\leq s} \int_\Omega (D^\alpha u)^2 dx$	Sobolev norm on H^s
$(u,v)_{H^s} = \int \sum_{\|\alpha\|\leq x} (D^\alpha u)(D^\alpha v) dx$ for u, v $\in$ H^s	inner product space
$f(v) = (f,v)$	bounded linear functional on H^1
$\|\|\|u\|\|\|^2 = \|u\|_a^2 = a(u,u)$	energy norm
V	any closed subspace of a Hilbert space
$V_h \subset V$	finite-dimensional subspace of V
u	solution
u_j	finite element approximation
u_j^*	iterative approximation

$\Omega \subset \mathcal{R}^d$	polygonal Lipschitz domain
$P_1(\cdot)$	linear functions on a domain
B_{j,m_j}	iterative operator on level j with m_j iterations
$D^\alpha u = (\partial/\partial x_1)^{\alpha_1}(\partial/\partial x_2)^{\alpha_2}\cdots(\partial/\partial x_d)^{\alpha_d}$	Partial derivative of order k where $\alpha = (\alpha_1, \ldots, \alpha_d)$, $\alpha_i \geq 0$, and $\lvert\alpha\rvert = \alpha_1 + \cdots + \alpha_d = k$
$C^m(\Omega)$	the linear space consisting of all functions u with partial derivatives $D^\alpha u$ of orders $0 \leq \lvert\alpha\rvert \leq m$ continuous on Ω
$C^\infty(\Omega) = \bigcap_{m=0}^{\infty} C^m(\Omega)$	the linear space of functions infinitely differentiable on Ω
$C_0^m(\Omega), C_0^\infty(\Omega)$	linear subspaces of $C^m(\Omega)$ and $C^\infty(\Omega)$, respectively, consisting of those functions that have compact support in Ω; that is, $u \in C_0^m(\Omega)$ implies that supp $u \equiv \overline{\{x \in \Omega : u(x) \neq 0\}} \subset\subset \Omega$
$\Omega_1 \subset\subset \Omega_2$	$\bar{\Omega}_1 \subset \Omega_2$ and $\mathrm{dist}(\bar{\Omega}_1, \partial\Omega_2) > 0$
$C^m\bar{\Omega}$	the linear space consisting of all functions u in $C^m(\Omega)$ for which $D^\alpha u$ is bounded and uniformly continuous on Ω for $0 \leq \lvert\alpha\rvert \leq m$. This space is a Banach space when equipped with the norm $\lVert u\rVert_{C^m(\bar{\Omega})} = \max_{0\leq\lvert\alpha\rvert\leq m} \sup_{x\in\Omega} \lvert D^\alpha u(x)\rvert$

$L^p(\Omega)$	the linear space of all equivalence classes of measurable linear functions u for which $\int_\Omega \lvert u(x)\rvert^p dx < \infty$
$\lVert u\rVert_{L^p(\Omega)} = \left[\int_\Omega \lvert u(x)\rvert^p dx\right]^{1/p}, \quad 1 \le p \le \infty$	When equipped with this norm, the spaces $L^p(\Omega)$ are Banach spaces. When $1 < p < \infty$, they are reflexive and for $p = 2$, $L^2(\Omega)$ is a Hilbert space with inner product $(u,v)_{L^2(\Omega)} = \int_\Omega u(x)\overline{v(x)}dx$ where $\overline{v(x)}$ is the complex conjugate of $v(x)$.
$\lVert u\rVert_{L^1(\Omega)} = \int_\Omega \lvert u(x)\rvert dx$	
$\lVert u\rVert_{L^\infty(\Omega)} = \operatorname*{ess\,sup}_{x\in\Omega}\lvert u(x)\rvert$	
$W^{m,p}(\Omega) = \{u \mid D^\alpha u \in l^p(\Omega) \text{ for } 0 \le \lvert\alpha\rvert \le m\}$	For integer $m \ge 0$ and $1 < p < \infty$, the Sobolev space $W^{m,p}(\Omega)$ of order (m,p) is the linear space of functions in $L^p(\Omega)$ whose distributional derivatives $D^\alpha u$ of all orders $\lvert\alpha\rvert$ such that $0 \le \lvert\alpha\rvert \le m$ are in $L^p(\Omega)$.
$\lVert u\rVert_{m,p,\Omega} = \left[\int_\Omega \sum_{\lvert\alpha\rvert\le m} \lvert D^\alpha u(x)\rvert^p dx\right]^{1/p} = \left[\sum_{\lvert\alpha\rvert\le m} \lVert D^\alpha u(x)\rVert^p_{L^p(\Omega)} dx\right]^{1/p}$	for $1 \le p < \infty$

$\|u\|_{m,\infty,\Omega} = \max_{\lvert\alpha\rvert\le m} \|D^\alpha u(x)\|_{L^\infty(\Omega)}$	Sobolev norms
$W_0^{m,p}(\Omega)$	the closure of $C_0^\infty(\Omega)$ in $W^{m,p}(\Omega)$
$H^m(\Omega) = W^{m,2}(\Omega)$	separable Hilbert space with inner product $(u,v)_m = \int_\Omega \sum_{\lvert\alpha\rvert\le m} D^\alpha u \overline{D^\alpha v} dx$
$H_0^m(\Omega) = W_0^{m,2}(\Omega)$	
$W^{0,p}(\Omega) = W_0^{0,p}(\Omega) = L^p(\Omega)$	for $1 \le p \le \infty$
$\lvert u\rvert_{m,p,\Omega} = \left[\int_\Omega \sum_{\lvert a\rvert=m} \lvert D^\alpha u(x)\rvert^p dx\right]^{1/p}$	seminorm defined on $W^{m,p}(\Omega)$
coercive (or elliptic or V-elliptic)	A bilinear form $a(\cdot,\cdot)$ on a normed linear space H is *coercive* on $V \subset H$ if there exists $\alpha > 0$ such that $a(v,v) \ge \alpha\|v\|_H^2$ for all $v \in V$. [5]
$(u,v)_k := (u,v)_{H^k(\Omega)} := \sum_{\lvert\alpha\rvert\le k} (D^\alpha u, D^\alpha v)_{L^2(\Omega)}$	norm on Hilbert space $H^k(\Omega)$ [15]
$\lvert u\rvert_k := \|u\|_{H^k(\Omega)} := \sqrt{\sum_{\lvert\alpha\rvert\le k} \|D^\alpha u\|_{L^2(\Omega)}^2}$	Sobolev norm [15]
$\lvert u\rvert_{k,0} := \sqrt{\sum_{\lvert\alpha\rvert=k} \|D^\alpha u\|_{L^2(\Omega)}^2}$	Sobolev norm [15]
$\lvert u\rvert_0 := \|u\|_{L^2(\Omega)} := \sqrt{\int_\Omega \lvert u(x)\rvert^2 dx}$	Sobolev norm [15]

admissible triangulation	a triangulation in which each pair of triangles has either no common points or a common vertex or a common side [5]
conforming element	one which satifies both continuity and completeness [5]
$\mathcal{T}$	triangulation
T	triangle
t	time step
$\mathbf{t}$	$\sum \Delta t_n$, total time
consistent	A finite difference method is consistent if the error approaches zero as $h \to 0$.
stability	An algorithm is stable if small changes in the original data result in small changes in the final results. [8]

Bibliography

[1] R. E. Bank and C. C. Douglas. Sharp estimates for multigrid rates of convergence with general smoothing and acceleration. *SIAM J. Numer. Anal.*, 22:617–633, 1985.

[2] R. E. Bank and T. Dupont. An optimal order process for solving elliptic finite element equations. *Math. Comp.*, 36:35–51, 1981.

[3] F. A. Bornemann and P. Deuflhard. The cascadic multigrid method for elliptic problems. *Numer. Math.*, 75:135–152, 1996.

[4] F. A. Bornemann and R. Krause. Classical and cascadic multigrid - a methodical comparison. In *Proceedings of the Ninth International Conference on Domain Decomposition*, pages 64–71, Bergen, Norway, 1998. Domain Decomposition Press.

[5] D. Braess. *Finite Elements. Theory, Fast Solvers and Applications in Solid Mechanics.* Cambridge University Press, Cambridge, 1997.

[6] D. Braess and W. Dahmen. A cascadic multigrid algorithm for the Stokes equation. *Numer. Math.*, 82:179–191, 1999.

[7] S. C. Brenner and L. R. Scott. *The Mathematical Theory of Finite Element Methods.* Texts in Applied Mathematics. Springer–Verlag, New York, 1994.

[8] R. L. Burden and J. D. Faires. *Numerical Analysis, Sixth Edition.* Brooks-Cole, New York, 1997.

[9] P. G. Ciarlet. *The Finite Element Method for Elliptic Problems.* North–Holland, Amsterdam, New York, 1978.

[10] P. Deuflhard. Cascadic conjugate gradient methods for elliptic partial differential equations: algorithm and numerical results. In *Domain Decomposition Methods in Scientific and Engineering Computing: Proceedings of the Seventh International Conference on Domain Decomposition*, volume 180 of *Contemporary Mathematics*, pages 29–42, Providence, Rhode Island, 1994. American Mathematical Society.

[11] P. Deuflhard, P. Leinen, and H. Yserentant. Concepts of an adaptive hierarchical finite element code. *Impact Comput. Sci. Eng.*, 1:3–35, 1989.

[12] S. H. Friedberg, A. J. Insel, and L. E. Spence. *Linear algebra.* Prentice–Hall, Englewood Cliffs, NJ, 1979.

[13] R. Haberman. *Elementary Applied Partial Differential Equations with Fourier Series and Boundary Value Problems.* Prentice–Hall, Upper Saddle River, NJ, 1998.

[14] W. Hackbusch. *Multigrid Methods and Applications*, volume 4 of *Computational Mathematics.* Springer–Verlag, Berlin, 1985.

[15] W. Hackbusch. *Elliptic Differential Equations: Theory and Numerical Treatment.* Springer-Verlag, New York, 1992.

[16] M. R. Hestenes. The conjugate-gradient method for solving linear systems. In *Proceedings of the Sixth Symposium in Applied Mathematics*, pages 83–102, Providence, Rhode Island, 1956. American Mathematical Society.

[17] J. M. Hyman. Mesh refinement and local inversion of elliptic differential equations. *J. Comput. Phys.*, 23:124–134, 1977.

[18] J. Jaffré and J. Roberts. Problem posed at IMA summer graduate program in mathematics, 2002.

[19] L. Johnson and R. Riess. *Numerical Analysis, Second Edition.* Addison-Wesley, Reading, MA, 1982.

[20] B. D. Reddy. *Introductory Functional Analysis with Applications to Boundary Value Problems and Finite Elements.* Springer–Verlag, New York, 1998.

[21] V. V. Shaidurov. Some estimates of the rate of convergence for the cascadic conjugate-gradient method. *Comput. Math. Appl.*, 31:161–171, 1996.

[22] Z. Shi and X. Xu. Cascadic multigrid method for elliptic problems. *East-West J. Numer. Math.*, 3:199–209, 1999.

[23] Z. Shi and X. Xu. Cascadic multigrid for parabolic problems. *J. Comput. Math.*, 18:551–560, 2000.

[24] R. S. Varga. *Matrix Iterative Analysis.* Prentice–Hall, Englewood Cliffs, NJ, 1962.

[25] H. Yserentant. Hierarchical bases. In *ICIAM'91: Proceedings of the Second International Congress on Industrial and Applied Mathematics*, pages 256–276. SIAM, Philadelphia, 1992.

[26] J. Zhang. Fast and high accuracy multigrid solution of the three dimensional Poisson equation. *J. Comput. Phys.*, 143:449–461, 1998.

VITA

Date of Birth

Born September 21, 1966. Mineola, New York.

Education

Master of Arts in Mathematics, University of Kentucky, May 1998.

Master of Education in Secondary Mathematics, Harvard University, June 1994.

Master of Science in Accounting, University of Hartford, August 1989.

Bachelor of Arts, Dartmouth College, June 1988, cum laude.

Publication

C. C. Douglas and K. Walters. Variable Order Cascadic Multigrid Algorithms, in Proceedings of ECCOMAS 2000, Barcelona, Spain, 2000.

Karen Munsell Walters