DOMAIN DECOMPOSITION AND HIGH ORDER DISCRETIZATION OF ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS

by

George G. Pitts

Dissertation submitted to the faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Mathematics

APPROVED:

C. J. Ribbens, Co-chairman

D. L. Russell, Co-chairman

L. W. Johnson

L. T. Watson

C. A. Beattie

November, 1994

Blacksburg, Virginia

DOMAIN DECOMPOSITION AND HIGH ORDER DISCRETIZATION OF ELLIPTIC PARTIAL DIFFERENTIAL EQUATIONS

by

George G. Pitts

Committee Co-chairmen:

C. J. Ribbens

Department of Computer Science

and

D. L. Russell

Department of Mathematics

(ABSTRACT)

Numerical solutions of partial differential equations (PDEs) resulting from problems in both the engineering and natural sciences result in solving large sparse linear systems $Au = b$. The construction of such linear systems and their solutions using either direct or iterative methods are topics of continuing research. The recent advent of parallel computer architectures has resulted in a search for good parallel algorithms to solve such systems, which in turn has led to a recent burgeoning of research into domain decomposition algorithms. Domain decomposition is a procedure which employs subdivision of the solution domain into smaller regions of convenient size or shape and, although such partitionings have proven to be quite effective on serial computers, they have proven to be even more effective on parallel computers.

Recent work in domain decomposition algorithms has largely been based on second order accurate discretization techniques. This dissertation describes an algorithm for the numerical solution of general two-dimensional linear elliptic partial differential equations with variable coefficients which employs both a high order accurate discretization and a Krylov subspace iterative solver in which a preconditioner is developed using domain decomposition. Most current research into such algorithms has been based on symmetric systems; however, variable PDE coefficients generally result in a nonsymmetric A, and less

is known about the use of preconditioned Krylov subspace iterative methods for the solution of nonsymmetric systems.

The use of the high order accurate discretization together with a domain decomposition based preconditioner results in an iterative technique with both high accuracy and rapid convergence. Supporting theory for both the discretization and the preconditioned iterative solver is presented. Numerical results are given on a set of test problems of varying complexity demonstrating the robustness of the algorithm. It is shown that, if only second order accuracy is required, the algorithm becomes an extremely fast direct solver. Parallel performance of the algorithm is illustrated with results from a shared memory multiprocessor.

ACKNOWLEDGEMENTS

I would like to thank my co-advisors Calvin J. Ribbens and David L. Russell for their guidance and encouragement. Without their assistance and support this work would not have been possible. I want to thank the other members of my advisory committee, Lee Johnson, Layne Watson, and Chris Beattie, for their willingness to serve on my committee and their time and valuable suggestions.

My special thanks go to Calvin Ribbens for his patience and perseverence. Dr. Ribbens provided me with both financial support and the topic of my research. Also, my special thanks to Matthew Heinkenschloss for his valuable help on the theoretical aspects of my research.

I am also grateful to the Department of Energy for the financial under grant DE-FG05-93ER25189 which made this research possible and the Argonne National Laboratories for the use of their computing facility.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

1.1 Historical Perspective

In what could be considered the discovery of partial differential equations (PDEs), in 1747 Jean Le Rond d'Alembert [115] formulated the hyperbolic wave equation

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \tag{1.1}$$

representing the small amplitude vibration of a taut string (c^2 is a positive constant related to flux and potential). D'Alembert felt his concept of modeling physical phenomena through such mathematical formulations was of equal importance to the discovery of the integral calculus, only less spectacular because of his use of words and symbols already known. In the mathematical modeling of the physical world, many other PDEs were formulated by scientists and mathematicians of the eighteenth and nineteenth centuries. The model parabolic equation (known as the heat equation)

$$\frac{\partial u}{\partial t} = c \frac{\partial^2 u}{\partial x^2} \tag{1.2}$$

was formulated by Fourier in 1807 [114]. In describing a steady state potential, the model elliptic equation (known as Laplace's equation though not originated by Laplace)

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{1.3}$$

was formulated as early as 1752 by Euler [115].

Until the early twentieth century such PDEs were mostly used as analytical tools in describing "real world" problems of engineering and the physical sciences. "Reputable" mathematicians sought simple analytic solutions while "applied" mathematicians seeking

numerical solutions were generally held in low esteem. Various by hand (pencil-and-paper) methods were known which approximated PDE solutions. During the late nineteenth and early twentieth centuries it became clear that most PDEs have no simple analytic solutions, and descriptions of numerical algorithms to solve PDEs began to appear in the literature. During this period very sophisticated mechanical calculators were developed which became quite popular. Numerical solution of PDEs representing scientific problems requires complex computations, and while any numerical calculation can in principal be performed either by hand or on a mechanical or electronic calculator, it was the advent of the modern digital computer which brought about a revolution in scientific computing. The digital computer enabled scientists and mathematicians to routinely make highly accurate complex calculations and thus computational mathematics became respectable.

1.2 Categories of PDEs

Because of their general applicability to common scientific problems, one of the most studied category of PDEs are two-dimensional second order linear equations of the form

$$Lu = g \tag{1.4}$$

where

$$Lu = au_{xx} + 2bu_{xy} + cu_{yy} + du_x + eu_y + fu \tag{1.5}$$

and where the coefficients a, b, c, d, e, f and g are functions of x and y. Such equations are *two-dimensional* since there are two independent variables, x and y (or perhaps x and t); they are *second order* since they include up to second order derivatives of the unknown function u; and they are *linear* since L is a linear operator in u. More complicated problems typically consist of a coupled system of two or more equations such as (1.4) in two or more unknowns; or the differential operator may be nonlinear, in which case numerical solution schemes often require that a series of problems such as (1.4) be solved.

To a large extent the second order derivatives in (1.5) determine the characteristics of such a PDE. Vichnevetsky [115] gives the following characterization. Consider the equivalent

Table 1.1: Categories of PDEs.

Category	$b^2 - ac$	Properties
Hyperbolic	> 0	There are two real families of characteristic lines on which solutions become unbounded.
Parabolic	$= 0$	There is one real family of characteristic lines on which solutions become unbounded.
Elliptic	< 0	There are no real characteristic lines on which solutions become unbounded.

matrix system

$$\begin{bmatrix} a & 2b & c \\ dx & dy & 0 \\ 0 & dx & dy \end{bmatrix} \begin{bmatrix} u_{xx} \\ u_{xy} \\ u_{yy} \end{bmatrix} = \begin{bmatrix} g - du_x - eu_y - fu \\ d(u_x) \\ d(u_y) \end{bmatrix}. \tag{1.6}$$

A unique solution exists if and only if the determinant of the coefficient matrix

$$\begin{vmatrix} a & 2b & c \\ dx & dy & 0 \\ 0 & dx & dy \end{vmatrix} = a\,dy^2 - 2b\,dx\;dy + c\,dx^2 \tag{1.7}$$

does not vanish (the second derivatives may thus be explicitly solved for). Lines on which the determinant vanishes are called *characteristic* lines. If we set

$$s = \frac{dy}{dx} \tag{1.8}$$

then the characteristic lines must satisfy

$$as^2 - 2bs + c = 0 \tag{1.9}$$

or

$$s = \frac{b \pm \sqrt{b^2 - ac}}{a}. \tag{1.10}$$

The value of the quantity under the radical determines three categories of solutions for Equation (1.5) as illustrated in Table 1.1.

1.3 Elliptic PDEs

Both hyperbolic and parabolic equations have characteristic lines on which solutions become unbounded while elliptic equations have no real characteristic lines and solutions remain bounded in compact domains. Elliptic equations represent physical systems in a state of equilibrium and arise whenever a physical system is in a steady state, and thus the time variable is not present. The value of the solution at each interior point depends on the value at all boundary points [115]. Two dimensional elliptic operators are of monotone type ($Lu \geq 0 \rightarrow u \geq 0$) ([8] [42] [16]) and have positive Green's functions ([8] [90]). If the PDE coefficients are continuous and $Lu \geq 0$ in the domain, then u can not have a local interior maximum unless u is constant and $Lu \equiv 0$. If $Lu \equiv 0$, then the maximum and minimum values appear on the boundary [6]. If u is an harmonic function ($\nabla^2 u = 0$) then the solution at each interior point is a weighted average of all the values of the solution on the boundary [124].

Solutions to elliptic equations are as smooth as their coefficient functions on the interior of the domain [8]. Thus if the coefficients and right hand side source term are all C^n functions, then so is the solution at any interior point [124]. Such smoothness properties make numerical computation of accurate solutions to elliptic problems easier than for non-elliptic PDEs [115]. Solutions of elliptic problems with analytic coefficients are analytic on the boundary wherever the boundary is analytic [8]. This is not the case on sharp edges (such as the corners of the unit square) at which point the derivatives do not exist. By the Cauchy-Kowalewski theorem [124], analyticity allows solutions to be continued across the boundary. Elliptic problems can be solved to any desired degree of accuracy by numerical methods [96] and the exact solutions can be considered simply as limits of the approximate solutions. Fast accurate elliptic PDE solvers have become important tools in many fields such as mechanics, electronics, and fluids.

We only consider well posed strictly elliptic equations whose domain is the unit square. A well posed problem is one in which the accompanying boundary conditions determine a

unique solution which depends continuously on the initial data. Strictly elliptic means that $b^2 - ac < 0$ for all x and y in the domain. Often the cross derivative term is not present and the required condition becomes $-ac < 0$, or equivalently, a and c have the same sign. We require that the solution be known and bounded on the domain boundary (a Dirichlet problem) and thus the solution remains bounded in the domain interior.

1.4 Numerical Solution of PDEs

Employing a digital computer to numerically solve a partial differential equation requires reducing the continuous problem into a discrete mathematical model of the problem. There are various such discretizations (including finite elements, finite differences, collocation, and others) which allow numerical solution of the PDE to whatever degree of accuracy is desired (subject to the precision available on any particular computer) [114].

Discretization methods typically overlay the domain with a *grid* or *mesh*, so that the approximate solution is based on a finite number of grid elements or points. The fineness of the mesh is usually measured by a parameter h which may represent a typical triangular element size or the average distance between successive grid lines in a rectangular mesh. The discretization results in a linear system $Au = b$ in which the coefficient matrix A has various properties. Desirable properties include A being sparse, banded, block tridiagonal, symmetric, diagonally dominanant, irreducible, of monotone type, an L-matrix, and positive definite. These properties are often required not only to obtain accurate direct solutions, but also to guarantee convergence of iterative linear system solvers.

While the resulting linear systems may be solved by a wide variety of direct solvers, such direct solution may not only be computationally expensive but also require a great deal of memory. Methods which require a fine grid to obtain the required accuracy result in very large linear systems. These large linear systems require substantial direct solve time (solve times increase geometrically with size) and large memory size. In addition, the characteristic fill-in of direct solvers precludes the use of efficient storage techniques.

1.4.1 High Order Discretization Methods

Given a rectangular mesh, a mesh size h, and an approximation u_h of u, we say that a method is a *kth order accurate*, or is a *kth order discretization*, or more simply is $\mathcal{O}(h^k)$ *accurate* if

$$|(u_h)_i - u_i| \leq Ch^k, \quad i = 1, \ldots, N \quad \text{for} \quad 0 < h < h_0 \tag{1.11}$$

for some h_0, where C is a positive constant independent of h and N is the number of mesh points. Large stencil, *HODIE*, and multigrid methods are three of the approaches which achieve higher order and/or faster solution times than classical finite difference methods.

The use of a higher order accurate method is an alternative way of obtaining higher accuracy without going to a finer grid. A higher order method requires substantially fewer grid points for a given accuracy resulting in smaller linear systems and much shorter direct solve times as illustrated in Table 1.2 (for comparison the solve time for a 100×100 system is considered unity).

Table 1.2: Comparison of $\mathcal{O}(h^2)$ and $\mathcal{O}(h^4)$ methods.

	$\mathcal{O}(h^2)$ Method		$\mathcal{O}(h^4)$ Method	
Error	Grid	Solve Time	Grid	Solve Time
10^{-4}	100×100	1	10×10	10^{-2}
10^{-8}	10000×10000	10^4	100×100	1

Table 1.2 is an idealized example in which an $n \times n$ grid results in an $N \times N$ linear system, with $N = n^2$ and $h = \frac{1}{n+1}$. A direct solve time of N^2 is realistic for an $N \times N$ banded system with bandwidth $\sqrt{N}$. The example also assumes that the error for the two methods is exactly h^2 and h^4, respectively. The $\mathcal{O}(h^4)$ method results in a linear system so much smaller than that of the $\mathcal{O}(h^2)$ method (with the same accuracy), that a direct solver may be suitable.

Large stencils add additional smoothness requirements to u, increase both the bandwidth and condition number of the linear system, and require even larger nonsymmetric stencils near boundaries to maintain a given order of accuracy. *HODIEX* achieves order $\mathcal{O}(N)$ solve

times and is quite easily parallelized without the disadvantages of large stencils. Multigrid slso achieves $\mathcal{O}(N)$ solves times but is not easily parallelized.

We will consider *HODIEX*, a high order discretization scheme which is an extension and generalization of the *ELLPACK* [100] discretization module *HODIE* (an acronym derived from "High Order Difference Approximation with Identity Expansion") of Lynch and Rice [89]. Both *HODIE* and *HODIEX* use a compact 3×3 stencil to discretize a general second order linear PDE with variable coefficients. They achieve a high order of accuracy on a wide range of problems. *HODIEX* differs from *HODIE* in that it discretizes problems with cross derivatives and allows more choices for the order of accuracy. *HODIE* methods determine the values of the approximate solution u_h of $Lu = g$ at grid points by requiring the approximation to be exact on a finite dimensional linear space. The achievable order of accuracy depends on the smoothness of both the solution and the PDE coefficients and on the dimension of the linear space. The *HODIEX* method and its theoretical properties are discussed at length in Chapters 2 and 3.

An advantage of *HODIE* methods, including *HODIEX*, is that they are based on a 3×3 compact stencil so that the coefficient matrix of the resulting linear system has a smaller bandwidth than if a larger stencil is used to achieve the same high accuracy. The coefficient matrix generated by *HODIE* methods also has other desirable features which are discussed in Chapter 2.

1.4.2 Linear System Solution Methods

As higher accuracy is demanded (equivalently as the number of grid points increases), higher order accurate discretization methods may also result in large linear systems. Direct solves of these systems may require too much time and memory, thus suggesting an iterative approach. Iterative solvers start from a first approximation and obtain progressively more accurate approximations to the solution using a sequence of steps. Iterative methods can use less memory than direct solvers by exploiting the sparsity of the coefficient matrix and avoiding fill-in. Generally only a matrix vector product Av is required and the coefficient

Table 1.3: Classical Methods.

Method	Iteration Matrix	Properties
Jacobi	$-D^{-1}(L+U)$	Although this method is very easy to use and program, it converges very slowly unless the iteration matrix B is strongly diagonally dominant. Although it parallelizes quite well it does not compete with newer polynomial or subspace methods.
Gauss-Seidel	$-(D+L)^{-1}U$	Faster convergence than Jacobi with strictly diagonally dominant or symmetric positive definite matrices. Multicolor ordering results in good parallelization. Equivalent to SOR with $\omega = 1$. Does not compete with newer polynomial or subspace methods.
SOR	$M^{-1}N$	$M = I + \omega D^{-1}L$ and $N = (1-\omega)I - \omega^{-1}D^{-1}U$. Accelerates convergence of Gauss-Seidel and often converges when Gauss-Seidel fails. Although it can parallelize quite well with multicolor orderings, it still does not compete with newer polynomial or subspace methods.

matrix need not be formed explicitly. For large linear systems iterative methods may also be faster than direct methods. Many iterative methods have been developed for various types of PDEs including classical methods and the more modern polynomial and subspace methods.

Classical Methods

Table 1.3 shows three classical methods and their properties. These methods have been around for many years and were first devised for pencil-and-paper or calculator computing. The first published accounts of such iterative methods are those of Gauss (1823) [113], Jacobi (1845) [122], Seidel (1874) [70], and the relaxation methods (SOR) of both Cross and Southwell (1930's) [109]. With these methods the coefficient matrix A is first decomposed as

$$A = L + D + U \tag{1.12}$$

where L is a strictly lower triangular matrix, D is a diagonal matrix, and U is a strictly upper triangular matrix. The general form of these methods is then

$$u_{j+1} = Bu_j + c \tag{1.13}$$

where various choices for the *iteration matrix* B are shown in Table 1.3.

The classical methods are convergent if and only if $\rho(B) < 1$, where $\rho(B)$ is the spectral radius of B (Birkhoff and Lynch [8]). Successive over relaxation (SOR) is an attempt to improve on slow convergence rates and depends on the computation of a relaxation parameter ω. It was only in the 1960's that David Young [121] presented a relatively straightforward algebraic method of choosing ω involving eigenvalue bounds of the iteration matrix B. There are also a variety of other methods including Picard/Richardson, SSOR (symmetric SOR), and ADI (alternating direction implicit). See Young [122], Varga [113], and Hageman & Young [70] for a more in depth treatment of classical iterative methods for solving the linear systems arising from PDE discretizations.

Polynomial and Subspace Methods

The development of new iterative methods parallels the development of the modern digital computer. Having computers of sufficient power and memory to actually solve a problem iteratively resulted in a resurgence of research into effective iterative methods. These methods differ from classical methods in that computations are more complex and often involve quantities that change at each iteration. Such changes would be difficult, if not impossible, with manual or calculator methods, but require little, if any, additional work on digital computers. Two areas of recent research are in polynomial methods and subspace methods.

Polynomial methods [58] start with an initial guess u_0 and generate a sequence of iterates u_j whose residuals satisfy

$$r_j = P_j(A)r_0 \tag{1.14}$$

where $P_j(A)$ is a polynomial in A. The classical methods described in Table 1.3 could be considered simple polynomial methods while Chebyshev Iteration is an example of a more complex polynomial method.

In the Chebyshev Iteration [93] method the P_j are determined in terms of Chebyshev

polynomials

$$P_j(z) = \frac{T_j\left(\frac{d-z}{c}\right)}{T_j\left(\frac{d}{c}\right)} \tag{1.15}$$

where T_j is the jth Chebyshev polynomial of the first kind. The parameters c and d depend on having good estimates of the spectrum of the coefficient matrix A. This method will often be used in conjunction with one of the subspace methods (described in the next paragraph) and a few iterations of a subspace method may provide the necessary information about the spectrum of A. Chebyshev Iteration avoids computation of inner products which generally is necessary for subspace methods. In practice the Chebyshev Iteration displays only linear convergence behavior, whereas subspace methods often display superlinear convergence [4].

Subspace methods compute an orthogonal sequence of vectors v_j which generate a sequence of finite dimensional subspaces

$$\mathcal{V}_1 \subset \ldots \subset \mathcal{V}_j \subset \mathcal{R}^N \tag{1.16}$$

where

$$\mathcal{V}_j = span\{v_1, v_2, \ldots, v_j\}. \tag{1.17}$$

In each iteration an approximation $u_j \in \mathcal{V}_j$ of u is computed. Subspace methods based on such an orthogonalization process were developed by a number of authors in the early 1950's.

Motivated by his search for solutions to eigenvalue problems, Lanczos [84] proposed a method in 1950 which was based on two mutually orthogonal vector sequences. In 1952, Lanczos [85] applied his method to solving linear systems. In 1952 Hestenes and Steifel [73] independently discovered the conjugate gradient (CG) method as a direct solver of symmetric positive definite (SPD) linear systems and stated that it had the following advantages over Gaussian elimination [73]:

- Less storage is required if the coefficient matrix A is sparse.
- There is no fill–in.

- As a direct solver, *CG* gives the solution in N steps (up to roundoff error).
- The error decreases each step.
- As an iterative method, *CG* gives adequate accuracy in less than N steps.
- *CG* requires no information on the spectrum of A.

The Hestenes and Steifel method is closely related to the Lanczos method in which the two mutually orthogonal sequences are reduced to one orthogonal sequence [4].

In 1951 Arnoldi [1] proposed a method with features of both the Lanczos and *CG* methods, but which did not require a symmetric matrix. Whereas both Lanczos and *CG* methods reduced a system to tridiagonal form, Arnoldi's method instead reduced it to upper Hessenberg form. Partly because as an iterative method *CG* was harder to program (using computers and languages of the time) than *SOR*, and as a direct method its roundoff errors were large and not self–correcting [60], the conjugate gradient method was originally used sparingly as a stand–alone method [8]; however, for the past twenty or more years it has been used effectively to accelerate convergence of a variety of methods. In 1971 Reid, in 1974 Barker, and in 1976 Golub [8] pointed out its effectiveness for this purpose. Since that time subspace methods have been an active research area with many new methods emerging. Table 1.4 summarizes some of the better known subspace methods.

GMRES

We will use the Krylov subspace method *GMRES* (generalized minimum residuals) of Saad and Schultz [103]. Krylov subspace methods construct a finite sequence of subspaces

$$\mathcal{V}_1 \subset \mathcal{V}_2 \subset \ldots \subset \mathcal{V}_j \subset \mathcal{R}^N \tag{1.18}$$

by an orthogonalization process where

$$\mathcal{V}_j = \mathcal{K}_j(A, v) \equiv span\{v, Av, \ldots, A^{j-1}v\}. \tag{1.19}$$

In each step an approximation u_j to u is constructed with $u_j \in \mathcal{V}_j$.

Table 1.4: Subspace Methods.

Method	Description	Properties
CG [73]	Conjugate Gradient	Used to solve SPD linear systems, its speed of convergence depends on the condition number of the coefficient matrix. Parallelization is limited by inner product calculations.
BiCG [61]	Biconjugate Gradient	Used to solve nonsymmetric systems in which the transpose of the coefficient matrix is available. Requires matrix vector multiplies of both the coefficient matrix and its transpose. Convergence may be oscillatory. Parallelization is again limited by inner product calculations.
QMR [61]	Quasi Minimum Residual	Designed to avoid oscillatory convergence behavior of *BiCG*. Although originally used to solve nonsymmetric systems in which the transpose of the coefficient matrix was available (as in *BiCG*), recently a transpose free version of *QMR* has been developed. Slightly higher computational costs and similar parallel properties as *BiCG*. When *BiCG* oscillates or diverges, *QMR* will still slowly reduce the residual.
CGS [108]	Conjugate Gradient Squared	Used to solve nonsymmetric systems and converges (or diverges) about twice as fast as *BiCG* while not requiring the matrix transpose. A major problem is that it tends to diverge if initial guess is "too good."
Bi-CGSTAB [117]	Biconjugate Gradient Stabilized	Used to solve nonsymmetric systems. While avoiding irregular convergence patterns of *CGS*, a loss of accuracy in the residual may occur. Does not require transpose operations.
GMRES [57]	Generalized Minimum Residual	Used to solve nonsymmetric systems, *GMRES* guarantees the smallest residual for a fixed number of iterations although steps can become increasingly expensive unless the method is restarted. Parallelization is limited by the number of inner products which increase with each step, and the global orthogonalization of vectors.

A major problem in Krylov subspace methods is the amount of memory required to store the j vectors which generate $\mathcal{K}_j(A, v)$. For this reason a maximum dimension is usually selected for *GMRES* at which point the iteration process is restarted using the last approximate as the new initial guess. We use full *GMRES* rather than restarted *GMRES* since the method converges rapidly and restarting is not necessary.

Krylov methods have several important features in regards to the coefficient matrix A of the linear system:

- A need not be formed explicitly, or if it is, efficient storage techniques may be used.
- There is no fill-in.

- A need not be symmetric (for *GMRES*; some solvers do require symmetry).
- Only the matrix vector product Av is required.

1.4.3 Preconditioning

Convergence of iterative methods in general, and subspace methods in particular, depends on the spectral properties of the coefficient matrix. Without a good preconditioner, slow convergence (or no convergence) may result in even longer computation time than direct solvers. A preconditioner transforms the problem into one with the same solution but with an iteration matrix which has more favorable spectral properties. For example, to solve the linear system

$$Au = b \tag{1.20}$$

we may solve instead the system

$$P^{-1}Au = P^{-1}b \tag{1.21}$$

which has the same solution as the original system; however, $P^{-1}A$ may have more favorable spectral properties. This is referred to as left preconditioning. In a similar manner one may construct a right preconditioned system

$$AP^{-1}v = b \tag{1.22}$$

where $v = Pu$. Other (symmetric) preconditioners can be defined by multiplying A on both the right and left. In general, a good preconditioner is based on an approximation to A or A^{-1}. When approximating A we want the preconditioner to be easily factorable so its inverse action is easy to compute. In approximating A^{-1} only multiplication by the preconditioner is required.

Cost of Preconditioning

Preconditioning incurs both more memory and additional computation costs. Depending on the preconditioner, extra memory costs are generally offset by the savings in memory

that result when using an iterative solver. The preconditioner must also be initially constructed and/or factored and applied for each iteration; thus there is a trade off between preconditioning and gain in convergence speed. The parallelization of both the construction and the application of the preconditioner is also an important issue.

The construction phase for some preconditioners may be computationally costly (such as incomplete factorizations) while other preconditioners need no construction at all (such as SSOR preconditioners). The preconditioner proposed in this dissertation is based on domain decomposition.

Domain Decomposition

Domain decomposition (DD) may be viewed as a type of "divide and conquer" [95] in which a problem is divided into smaller pieces and then the results pasted together. Domain decomposition may be used in solving PDEs arising from finite element, finite difference, or other discretization methods. The general idea is to divide the domain into smaller, more manageable pieces. By "manageable" one might mean a simpler geometry or differential equation in each subdomain, a smaller problem size in each subdomain (fewer grid points), a more favorable matrix structure (block diagonal, smaller bandwidth, etc.), better parallelization (subdomains solved simultaneously), faster solution times (serial or parallel), or any combination of these and other ideas that make the global problem somehow "easier" to solve. Although there are various ways in which domain decomposition may be used, we will only consider DD as a method of constructing an efficient parallel preconditioner for the iterative solver.

The two approaches to decomposing a domain are overlapping subdomains and non-overlapping subdomains [40]. One of the first domain decomposition algorithms is credited to Schwarz who proposed overlapping subdomains in 1869 [95]. The advent of parallel computers led to a resurgence in research into domain decomposition algorithms. DD algorithms generally parallelize well, so well in fact that we often hear the term "embarassingly parallel" applied to them. In 1986 Bjorstad and Widlund [12] proposed non-overlapping

subdomains in an algorithm to solve elliptic problems and Bramble, Pasciak, and Schatz [26] proposed the use of domain decomposed preconditioners. In 1987 Chan [37] did an analysis on domain decomposition preconditioners and the first of annual international symposiums on domain decomposition methods was held in Paris [62].

A simple non-overlapping domain decomposition is accomplished as follows. Assume a rectangular mesh has been placed over the global domain. A set of rectangular subdomains is formed by drawing horizontal and/or vertical lines called *interfaces* across the domain by connecting sets of horizontal or vertical mesh points. The mesh points contained within each of these rectangular areas, not including mesh points on interface lines (the boundaries of the rectangular areas), are known as *interior points* and the set of points contained in one of these rectangular areas is known as an *interior subdomain.* Points on only one interface line are called *interface points.* Points on the intersection of two interface lines are called *crosspoints.* These point sets form *interface subdomains* and a *crosspoint subdomain,* respectively, in a manner to be fully described in Chapter 4.

The proposed domain decomposed preconditioner P is constructed to approximate A by exactly matching A in the interior subdomains, while only approximating A in the interface and crosspoints subdomains. This decomposed domain approach to the construction of the preconditioner P requires only modest additional computation time (most of the entries of the preconditioner P are the same as in A) and results in a preconditioner which is easily factored and for which application of P^{-1} is a highly parallel operation.

1.5 Summary

The algorithm described in this dissertation combines a discretization method which achieves high order on a large class of linear two dimensional variable coefficient elliptic PDEs, a preconditioner based on domain decomposition which may be applied with a minimal amount of computation time and parallelizes well, and a preconditioned iterative solver which exploits the sparsity and other special structure of the coefficient matrix. The end result is a method which converges rapidly with high accuracy on a wide range of

problems, and which executes efficiently on a parallel computer.

The remainder of this dissertation is organized as follows. Chapter 2 is a description of *HODIEX* with details on finding discrete linear operators which approximate L. Implementation and computational considerations are also presented. Results on the direct sequential solution of a suite of test problems are presented showing the order of accuracy achieved. Chapter 3 summarizes the supporting theory for *HODIEX*.

Chapter 4 is a detailed description of our domain decomposed preconditioner. Details on the construction of subdomains for several decompositions are presented along with a description using matrix notation. Chapter 5 presents supporting theory of the preconditioned solver for the *HODIEX-DD-GMRES* method. The rate of convergence is characterized in terms of the general theory supporting *GMRES*, the fact that the preconditioner is a good approximation of A^{-1}, and the fact that the preconditioner is also a lower order discretization of the PDE (resulting in a good initial guess).

Chapter 6 presents performance on a selection of test problems using the preconditioned iterative solver with both second and fourth order accurate discretizations. The results show that one form of the preconditioner is optimal in the sense that a specified constant ratio of coarse grid size to fine grid size results in a constant number of iterations, independent of the number of grid points. With an appropriate domain decomposition, the preconditioner alone is shown to be an extremely fast $\mathcal{O}(h^2)$ direct solver. Finally, Chapter 7 presents conclusions and other future research ideas.

Chapter 2

THE HODIEX DISCRETIZATION METHOD

The numerical solution of a PDE begins with a discretization process which converts a continuous infinite dimensional problem into a discrete finite dimensional problem. In this chapter, after a brief discussion of the discretization process, we present details of the *HODIEX* discretization method. *HODIEX* discretizes linear two dimensional elliptic PDEs with variable coefficients and achieves a high order of accuracy on a wide range of problems. The method is an extension of the *ELLPACK* [100] discretization module *HODIE* (an acronym derived from "High Order Difference Approximation with Identity Expansion") of Lynch and Rice [89] . Performance on several test problems is reported along with comparisons with lower order classical finite difference methods.

2.1 Problem Statement

We assume a domain $\mathcal{D} = (0,1) \times (0,1)$. Given bounded $a, b, c, d, e, f, g \in \mathcal{C}^2(\mathcal{D})$, the problem is to approximate the function $u(x,y)$ that satisfies the general linear second order partial differential equation

$$Lu \equiv au_{xx} + 2bu_{xy} + cu_{yy} + du_x + eu_y + fu = g \quad \text{in } \mathcal{D} \tag{2.1}$$

where $ac \neq 0$ and L a strictly elliptic operator ($b^2 - ac < 0$), and with Dirichlet boundary conditions $u = \phi \equiv 0$ on $\partial\mathcal{D}$.

2.2 The Discretization Process

The domain $\mathcal{D}$ is covered by three distinct sets of points as follows. A rectangular mesh comprised of n_x vertical and n_y horizontal interior lines is placed over $\mathcal{D}$. Mesh widths are

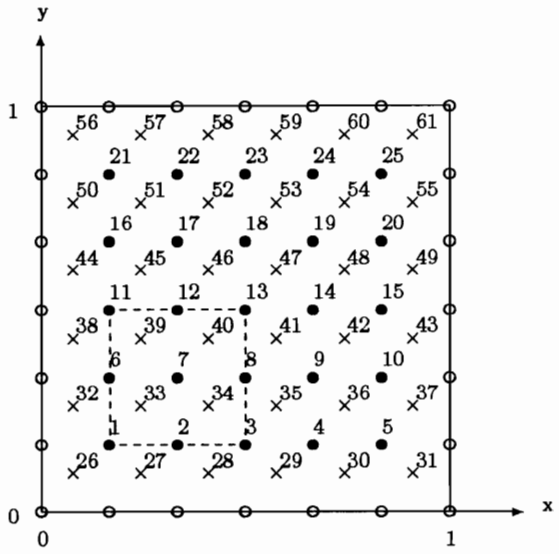


Figure 2.1: Grid points, evaluation points, and boundary points.

thus $h_x = \frac{1}{n_x+1}$ and $h_y = \frac{1}{n_y+1}$ in the x and y directions, respectively. For simplicity of notation we assume $h_x = h_y = h$, $n_x = n_y = n$, and $N = n \times n$.

The first set called *boundary points* is determined by the intersections of mesh lines with the domain boundary. The second set of points $\mathcal{G}$ called *grid points* is determined by the nodes of the mesh. $\mathcal{G}$ is the set of N points on which (2.1) will be discretized. The points in $\mathcal{G}$ are ordered using *natural ordering*, that is, the points are numbered from left to right and from bottom to top. Note that our convention assumes all grid points are in $\mathcal{D}$ and none on $\partial\mathcal{D}$.

The third set of points called *additional evaluation points* consists of non-grid points distributed throughout the domain. We denote by $\mathcal{E}$ the set of *evaluation points* consisting of both the grid points $\mathcal{G}$ and the additional evaluation points. The evaluation points are ordered with the first N grid points corresponding to the numbering in $\mathcal{G}$. This ordering is used as a *point index* to reference all non-boundary points. Figure 2.1 shows a typical domain with a 5×5 rectangular mesh. The 25 grid points are denoted by $\bullet$ and 36 additional evaluation points denoted by $\times$ (thus $|\mathcal{E}| = 61$).

Using the point index, we denote by p_i the (x, y) coordinates of the ith point for $i =$

$1, \ldots, |\mathcal{E}|$. The true solution u at the point p_i is $u_i = u(p_i)$ for $i = 1, 2, \ldots, N$ with a similar notation for g. The approximation of u on the N grid points is denoted by the N–vector u_h with the ith component $(u_h)_i$ denoting the approximation of the true solution u at p_i. We also define two functions r_h and s_h which map continuous functions into vectors. Thus if ψ is a function defined over the grid then

$$r_h\psi \equiv (\psi_1, \psi_2, \ldots, \psi_N)^T \tag{2.2}$$

or over the evaluation points then

$$s_h\psi \equiv (\psi_1, \psi_2, \ldots, \psi_{|\mathcal{E}|})^T. \tag{2.3}$$

Thus $r_h\psi$ is an N–vector and $s_h\psi$ is an $|\mathcal{E}|$–vector. The significance of s_h and the additional evaluation points will be made clear in the discussion of the *HODIEX* discretization method in Section 2.3. When discussing standard finite differences we have $|\mathcal{E}| = N$ and $r_h = s_h$.

2.2.1 Operator Discretizations and Stencils

The continuous differential operator L can be approximated by replacing each derivative with a discrete approximate. There are a number of such approximations; for example, central finite differences can be used to approximate derivatives of u at the grid point p_i with the following $\mathcal{O}(h^2)$ accurate formulas:

$$\begin{aligned}
u_{xy} &= \frac{1}{4h^2}(u_{i+n+1} - u_{i-n+1} - u_{i+n-1} + u_{i-n-1}) + \mathcal{O}(h^2) \\
u_{xx} &= \frac{1}{h^2}(u_{i+1} - 2u_i + u_{i-1}) + \mathcal{O}(h^2) \\
u_{yy} &= \frac{1}{h^2}(u_{i+n} - 2u_i + u_{i-n}) + \mathcal{O}(h^2) \\
u_x &= \frac{1}{2h}(u_{i+1} - u_{i-1}) + \mathcal{O}(h^2) \\
u_y &= \frac{1}{2h}(u_{i+n} - u_{i-n}) + \mathcal{O}(h^2).
\end{aligned} \tag{2.4}$$

Notice that a total of 9 grid points are used in these approximations. Figure 2.2 depicts these 9 points where the central grid point p_i is indicated by o.

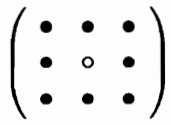


Figure 2.2: 9-point compact stencil.

We define a *computational element* to be the set of all evaluation points contained in the square determined by the four points $(x_i \pm h, y_i \pm h)$. $\mathcal{D}_i$ is the set of grid point indexes contained in the computational element centered at point i and $\mathcal{E}_i$ is the set of grid point indexes and additional evaluation point indexes contained in the computational element centered at point i. The cardinality of $\mathcal{D}_i$ is $|\mathcal{D}_i| = 9$ while $|\mathcal{E}_i| \equiv M$ depends on the selection of an appropriate linear space as discussed in Section 2.3. For example, in Figure 2.1 the dashed box indicates computational element 7 in which

$$\begin{aligned} \mathcal{D}_7 &= \{7, 8, 6, 12, 2, 13, 11, 1, 3\} \\ \mathcal{E}_7 &= \{7, 8, 6, 12, 2, 13, 11, 1, 3, 33, 34, 39, 40\}. \end{aligned}$$

In this case $|\mathcal{E}_7| = 13$. The order of the grid point indexes is always the central grid point first, followed by the other eight grid points, and the additional evaluation points last.

The finite difference discretization process entails replacing the continuous problem

$$Lu = g \tag{2.5}$$

with a discrete linear estimate

$$L_i r_h u \approx (Lu)_i = (g)_i \tag{2.6}$$

which approximates (2.5) at each grid point p_i for $i = 1, \ldots, N$. Generally L_i will be a linear combination of estimates such as those in (2.4). These approximations are assembled into a linear system

$$Au_h = r_h g \tag{2.7}$$

where u_h is the approximation of u at the grid points and the entries in each row of A represent the coefficients of the linear combinations of u_i.

The linear combination of derivatives used in the discrete approximation of Lu is pictorially represented by a diagram called a *stencil*. For example, the discrete approximation of Poisson's equation

$$u_{xx} + u_{yy} = g, \tag{2.8}$$

obtained from (2.4) is represented by the equation

$$\frac{1}{h^2}\begin{array}{|c|c|c|}\hline & 1 & \\ \hline 1 & -4 & 1 \\ \hline & 1 & \\ \hline\end{array} \quad u = g + \mathcal{O}(h^2). \tag{2.9}$$

Numbers inside the stencil represent the coefficients of the corresponding values of u. The constant $\frac{1}{h^2}$ indicates multiplication of each of the stencil entries. Equation (2.9) represents the approximation

$$\frac{1}{h^2}(u_{i+n} + u_{i-1} - 4u_i + u_{i+1} + u_{i-n}) = g_i + \mathcal{O}(h^2). \tag{2.10}$$

where $i = 1, \ldots, N$.

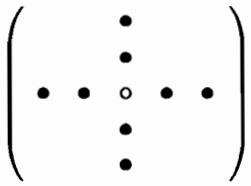


Figure 2.3: The 9-point star.

In stencil Equation (2.9) the indices for u and g are not shown, but it is understood that the equation indicates evaluation of u and g at the points indicated by Equation (2.10). Also, if the central grid point p_i is adjacent to a boundary, it is understood that known values of ϕ are used at neighboring boundary points. If a stencil entry is 0 then, for simplicity, it is left blank. The corner points of the stencil in (2.9) are blank since, with no cross derivatives, the entries are 0. Square stencils, such as that shown by Figure 2.2, are called *compact* stencils. An example of a *non-compact stencil* is the 9–point star shown in Figure 2.3.

The Poisson approximation shown in (2.9) is only $\mathcal{O}(h^2)$ accurate as are all approximations using formulas (2.4). Approximations using these central finite difference formulas will be exact whenever u is a polynomial of degree three, or of degree two if there are first order derivative terms in Lu.

2.2.2 Solution of the Discrete Problem

Finite difference discretization techniques generate one equation for each interior grid point. Boundary conditions may generate additional equations; however, in this dissertation we assume Dirichlet boundary conditions. With Dirichlet boundary conditions, if the discretization of a grid point contains boundary points, the values at the boundary points are translated to the right hand side of the linear system. In this dissertation we have assumed that $u = \phi \equiv 0$ on the boundary and thus these terms are zero.

The coefficient matrix A of Equation (2.7) has properties which play an important role in the solution of the linear system. We will use the following definitions (Varga [113]):

Definition 2.1 *A is said to be (weakly) diagonally dominant if $|a_{ii}| \geq \sum_{j \neq i} |a_{ij}|$, $i = 1,\ldots,N$, with inequality for at least one i.*

Definition 2.2 *A is said to be positive definite if $x^T Ax > 0$ for all $x \neq 0$.*

Definition 2.3 *A is said to be an L-matrix if $a_{ii} > 0$, $i = 1, \ldots, N$ and $a_{ij} \leq 0$, $i \neq j$.*

Definition 2.4 *A is said to be of monotone type if $Ax \geq 0$ implies $x \geq 0$ (A^{-1} is monotone).*

Definition 2.5 *A is said to be reducible if there are permutation matrices P and Q such that*

$$PAQ = \begin{pmatrix} A_1 & A_3 \\ 0 & A_4 \end{pmatrix}$$

where A_1 and A_4 are square matrices. If no such permutation matrices P and Q exist, then A is said to be irreducible.

A number of these properties are associated with using formulas (2.4) to discretize elliptic PDEs. For sufficiently small mesh size, A is weakly diagonally dominant (Young [122]). A is irreducible and of monotone type (Birkhoff and Lynch [8]). A is an L-matrix and thus is positive definite (Varga [113]). If the PDE coefficients are constant then A is symmetric and all eigenvalues in both A and its inverse are positive. These properties not only guarantee solution using direct solvers, but also make the linear systems amenable to iterative solvers. Positive definiteness assures no pivoting is required for direct solvers (Golub and Van Loan [63]) and much of the theory on iterative methods applies to a matrix with these properties (Young [122] and Varga [113]).

The linear systems associated with using 3×3 compact stencils have a number of desirable computational properties. The coefficient matrix A is sparse with at most 9 nonzeros per row. The use of natural ordering results in a banded system with a bandwidth of $n + 1$ in which A is block tridiagonal.

2.3 HODIEX

Standard finite difference approximations give the highest ("optimal") order of accuracy obtainable using a 3×3 compact stencil for *general* second order linear PDEs [8]. However, for certain operators, a higher order is possible. For example, the Laplacian (which is symmetric and invariant under rotations) has the $\mathcal{O}(h^6)$ approximation [42]

$$\nabla^2 u = \frac{1}{6h^2} \begin{array}{|c|c|c|} \hline 1 & 4 & 1 \\ \hline 4 & -20 & 4 \\ \hline 1 & 4 & 1 \\ \hline \end{array} u + \mathcal{O}(h^6). \tag{2.11}$$

High accuracy using an $\mathcal{O}(h^2)$ discretization method requires the use of a very fine grid. Fine grids lead to very large linear systems and correspondingly large direct solve times. Higher orders of accuracy may also be achieved on coarse grids by using larger stencils such

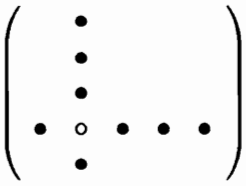


Figure 2.4: 9-point nonsymmetric star.

as is shown by the following 9-point star stencil for Poisson's equation

$$\frac{1}{h^2}\begin{array}{|c|c|c|c|c|}\hline & & -1 & & \\ \hline & & 16 & & \\ \hline -1 & 16 & -60 & 16 & -1 \\ \hline & & 16 & & \\ \hline & & -1 & & \\ \hline\end{array} \quad u = g + \mathcal{O}(h^4) \tag{2.12}$$

which is $\mathcal{O}(h^4)$ accurate as indicated.

Although the use of larger stencils results in increased accuracy, they also result in the loss of some of the desirable features of 3×3 compact stencils. The 9-point star shown in Figure 2.3 increases the bandwidth to $2n$ on interior points. Handling grid points near the boundary requires larger nonsymmetric stars (to avoid extending outside the boundary and to maintain a given accuracy). Figure 2.4 shows a nonsymmetric star for the bottom left corner grid point in which 9 points are required to maintain $\mathcal{O}(h^4)$ accuracy for the Laplacian (up to 13 points may be required for more general operators).

Large nonsymmetric stars add additional smoothness requirements to u. They also result in an increase to the bandwidth (to $3n$ for the 9-point star in Figure 2.4). Computationally, the bandwidth can be maintained at $2n$ by the early elimination of some variables in a process known as *static condensation* [70] as the linear system is constructed, but this results in an increase in the condition number of the linear system and a possible loss of definiteness [10].

2.3.1 High Order Accuracy Using Compact Stencils

In the preceding discussion of finite differences, the *source term* g (the right hand side of the PDE) was only evaluated at the central stencil point. Collatz [42] demonstrated that more accurate approximations can be obtained by evaluating g at more than one point and using a weighted average of these values. His *Mehrstellenverfahren* (literally "more points method") combines Taylor series expansions of g at additional points with linear combinations of Taylor series expansions of u and Lu eliminating as many terms as possible. Lynch and Rice [89] derived the *HODIE* method which uses no derivatives of g and also achieves high order.

By requiring the approximation to be exact on higher dimensional linear spaces, *HODIE* achieves $\mathcal{O}(h^4)$ accuracy on general linear problems with no cross derivatives with orders of accuracy as high as $\mathcal{O}(h^6)$ on certain problems with constant coefficients. Our extended version of the *HODIE* code, which we refer to as *HODIEX*, achieves $\mathcal{O}(h^4)$ accuracy on general problems and also handles problems with cross derivatives. *HODIEX* can also achieve $\mathcal{O}(h^6)$ on many smooth problems with no requirement of constant coefficients. Smooth solution and coefficients make possible high orders of accuracy. For example, if $b = 0$ and $a = c = 1$ then at least 4th order accuracy may be obtained while if $b = 0$, $a = c$, $d_y = e_x$, and f =constant then 6th order accuracy may be achieved on Dirichlet problems such as are considered here (Young [122]).

All *HODIE* methods are based on a compact 9–point stencil and do not have the problems associated with using larger stencils for higher accuracy. An example of a fourth order approximation for Poisson's equation using *HODIEX* is

$$\frac{1}{6h^2}\begin{array}{|c|c|c|}\hline 1 & 4 & 1 \\ \hline 4 & -20 & 4 \\ \hline 1 & 4 & 1 \\ \hline\end{array}\ u = \begin{array}{|c|c|c|}\hline & 1 & \\ \hline 1 & 8 & 1 \\ \hline & 1 & \\ \hline\end{array}\ g + \mathcal{O}(h^4). \tag{2.13}$$

Notice that g is evaluated at the central stencil point and at four neighboring points.

2.3.2 Why HODIEX? A Simple One Dimensional Example

To understand how *HODIEX* works we will consider a simple one dimensional example. Consider (2.1) in which $a = 1$ and $b = c = d = e = f = 0$, yielding the ODE

$$u_{xx} = g. \tag{2.14}$$

Central finite differences yield the $\mathcal{O}(h^2)$ stencil equation

$$\frac{1}{h^2} \begin{array}{|c|c|c|} \hline 1 & -2 & 1 \\ \hline \end{array} \quad u = g + \mathcal{O}(h^2) \tag{2.15}$$

which represents the approximation

$$L_i r_h u \equiv \frac{1}{h^2}(u_{i+1} - 2u_i + u_{i-1}) = g_i + \mathcal{O}(h^2). \tag{2.16}$$

Equation (2.16) is exact if u is in the space $\mathcal{P}_3$ (polynomials with maximum total degree 3). To see this assume that

$$u = a + bx + cx^2 + dx^3.$$

At the point $x_i = ih$ we get

$$L_i r_h u = \frac{1}{h^2}(u_{i+1} - 2u_i + u_{i-1}) = 2c + 6dx_i = (u_{xx})_i = g_i.$$

We seek a higher order of accuracy than the $\mathcal{O}(h^2)$ accuracy of Equation (2.16) and consider an approximation exact on $\mathcal{P}_4$. To do this we define the operators L_i and I_i at grid point x_i by

$$L_i r_h u \equiv \alpha_1 u_{i+1} + \alpha_2 u_i + \alpha_3 u_{i-1} \tag{2.17}$$

$$I_i s_h g \equiv \beta_1 g_{i+1} + \beta_2 g_i + \beta_3 g_{i-1} \tag{2.18}$$

where α_1, α_2, α_3, β_1, β_2, and β_3 are undetermined coefficients which define the operators L_i and I_i. We choose the αs and βs so that

$$L_i r_h u = I_i s_h g + \mathcal{O}(h^k), \quad i = 1, \ldots, n \tag{2.19}$$

where k will be determined by making the approximation exact on some finite dimensional linear space. In other words, we want

$$L_i r_h v - I_h s_h(Lv) = 0 \tag{2.20}$$

at each grid point for all v in some space of polynomials. Equation (2.16) corresponds to Equation (2.19) exact on $\mathcal{P}_3$ yielding $k = 2$. To require Equation (2.19) to be exact on $\mathcal{P}_4$ we use the following approach.

Select as a basis for $\mathcal{P}_4$ centered at the point x_i (for notational convenience we show each basis element centered at the origin):

$$\{b_1, b_2, b_3, b_4, b_5\} = \left\{1, \frac{x}{h}, \frac{x^2}{h^2}, \frac{x(x^2-h^2)}{h^3}, \frac{x^2(x^2-h^2)}{h^4}\right\} \tag{2.21}$$

chosen so that at grid points $-h, 0$, and h we have $b_4 = b_5 = 0$. Requiring Equation (2.19) to be exact on $\mathcal{P}_4$ at x_i gives

$$L_i r_h b_k - I_i s_h(Lb_k) = 0, \quad k = 1, \ldots, 5. \tag{2.22}$$

This system of equations is homogeneous (right hand side is 0) so we add a normalization equation

$$\beta_1 + \beta_2 + \beta_3 = 1. \tag{2.23}$$

This yields the 6×6 linear system

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 1 & -\frac{2}{h^2} & -\frac{2}{h^2} & -\frac{2}{h^2} \\ 0 & 0 & 0 & \frac{6}{h^2} & 0 & -\frac{6}{h^2} \\ 0 & 0 & 0 & -\frac{10}{h^2} & \frac{2}{h^2} & -\frac{10}{h^2} \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}. \tag{2.24}$$

The choice of basis functions has reduced the system to the form

$$\begin{pmatrix} A_1 & B \\ 0 & C \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ r \end{pmatrix} \tag{2.25}$$

and the βs can be computed first by solving the system

$$C\beta = r.$$

Then the αs are computed by solving the system

$$A_1\alpha = -B\beta.$$

In this case the result is

$$\alpha_1 = \alpha_3 = \frac{1}{h^2}, \quad \alpha_2 = -\frac{2}{h^2}, \quad \beta_1 = \beta_3 = \frac{1}{12}, \quad \text{and } \beta_2 = \frac{10}{12}$$

at each grid point (because the coefficients are constant) and thus

$$L_i r_h u = \frac{1}{h^2} \boxed{1}\boxed{-2}\boxed{1}\, u \tag{2.26}$$

and

$$I_i s_h g = \frac{1}{12} \boxed{1}\boxed{10}\boxed{1}\, g \tag{2.27}$$

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

A Taylor series expansion verifies that $k = 4$ and

$$\frac{1}{h^2}(u_{i+1} - 2u_i + u_{i-1}) = \frac{1}{12}(g_{i+1} + 10g_i + g_{i-1}) + \mathcal{O}(h^4). \tag{2.28}$$

This equation is known as the Stormer-Numerov formula [91]. Requiring the approximation (2.19) to be exact on $\mathcal{P}_4$ instead of $\mathcal{P}_3$ has increased the order of accuracy from $\mathcal{O}(h^2)$ to $\mathcal{O}(h^4)$.

2.3.3 Construction of the HODIEX Equations

HODIEX is a generalization of the above procedure to two dimensions. From the preceding example it is clear that there are two point sets involved in the discretization of $Lu = g$ at a grid point p_i. At grid point i we define the discrete operator

$$L_i r_h u \equiv \sum_{j \in \mathcal{D}_i} \alpha_j u_j \tag{2.29}$$

where $\mathcal{D}_i$ is the grid point index set for computational element i. Similarly we define another discrete operator

$$I_i s_h g \equiv \sum_{j \in \mathcal{E}_i} \beta_j g_j \tag{2.30}$$

where $\mathcal{E}_i$ is the evaluation point index set for computational element i. Note that in the above notation the αs and βs have an implicit dependence on i and, except for a constant coefficient PDE, will vary at each grid point.

The coefficients determining the operators L_i and I_i are determined at each grid point by selecting a K dimensional linear space $\mathcal{P}$ with a basis $\{b_1, b_2, \ldots, b_K\}$ and requiring

$$L_i r_h b_k - I_i s_h (L b_k) = 0, \quad k = 1, \ldots, K. \tag{2.31}$$

Equation (2.31) results in a homogeneous linear system of K equations

$$\sum_{j \in \mathcal{D}_i} \alpha_j (b_k)_j - \sum_{j \in \mathcal{E}_i} \beta_j (L b_k)_j = 0, \quad k = 1, \ldots, K, \tag{2.32}$$

at each grid point p_i and so we add an additional normalization equation such as

$$\beta_{j_1} = 1 \tag{2.33}$$

where j_1 is the index of the central stencil point in $\mathcal{E}_i$, or

$$\sum_{j \in \mathcal{E}_i} \beta_j = 1. \tag{2.34}$$

The system of equations (2.32) is referred to as the *HODIEX equations* and Equation (2.33) or (2.34) is referred to as a *normalization* equation. With $M = |\mathcal{E}_i|$ evaluation points, the system represented by (2.32) and (2.34) has the more explicit form

$$\begin{pmatrix} (b_1)_{j_1} & \cdots & (b_1)_{j_9} & -(Lb_1)_{j_1} & \cdots & -(Lb_1)_{j_M} \\ (b_2)_{j_1} & \cdots & (b_2)_{j_9} & -(Lb_2)_{j_1} & \cdots & -(Lb_2)_{j_M} \\ (b_3)_{j_1} & \cdots & (b_3)_{j_9} & -(Lb_3)_{j_1} & \cdots & -(Lb_3)_{j_M} \\ \vdots & \ddots & \cdots & \vdots & \ddots & \cdots \\ (b_K)_{j_1} & \cdots & (b_K)_{j_9} & -(Lb_K)_{j_1} & \cdots & -(Lb_K)_{j_M} \\ 0 & \cdots & 0 & 1 & \cdots & 1 \end{pmatrix} \begin{pmatrix} \alpha_{j_1} \\ \vdots \\ \alpha_{j_9} \\ \beta_{j_1} \\ \vdots \\ \beta_{j_M} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}. \tag{2.35}$$

The K basis functions and one normalization equation result in a total of $K+1$ rows while the M βs and 9 αs result in a total of $M+9$ columns. Choosing $M+9=K+1$ ($M=K-8$) as the number of evaluation points results in a square system, and, if the system has full rank, it is easy to solve. In some situations, however, the system (2.35) may be singular, though still consistent, in which case care must be taken in finding a solution. We will discuss under what circumstances the *HODIEX* equations are solvable in the next chapter.

2.3.4 Solution of the HODIEX Equations

The system of *HODIEX* equations can be made reducible allowing separate solves for the αs and βs by a suitable choice of basis functions. For example, consider the space $\mathcal{P}_4$ of polynomials with maximum total degree 4. There are a total of $K=\frac{(4+1)(4+2)}{2}=15$ basis functions. The basis functions are dependent on the grid point p_i, and for notational clarity we translate them from p_i to $(0,0)$. We select as the first 9 basis functions

$$\begin{array}{lll} b_1=1 & b_4=\frac{x^2}{h_x^2} & b_7=\frac{x^2y}{h_x^2h_y} \\ b_2=\frac{x}{h_x} & b_5=\frac{xy}{h_xh_y} & b_8=\frac{xy^2}{h_xh_y^2} \\ b_3=\frac{y}{h_y} & b_6=\frac{y^2}{h_y^2} & b_9=\frac{x^2y^2}{h_x^2h_y^2}. \end{array} \tag{2.36}$$

These first 9 basis elements have been chosen so that their value is 0 or ± 1 for points in $\mathcal{D}_i$. The remaining basis elements b_{10} through b_K are selected to make the linear system reducible (as in the example above) by choosing basis elements that vanish at points in $\mathcal{D}_i$. This is accomplished by each basis function having a factor $x\left(x^2-h_x^2\right)$ or $y\left(y^2-h_y^2\right)$. The remaining basis functions for $\mathcal{P}_4$ are

$$\begin{array}{ll} b_{10}=\frac{x\left(x^2-h_x^2\right)}{h_x^3} & b_{13}=\frac{y\left(y^2-h_y^2\right)}{h_y^3} \\ b_{11}=\frac{x^2\left(x^2-h_x^2\right)}{h_x^4} & b_{14}=\frac{xy\left(y^2-h_y^2\right)}{h_xh_y^3} \\ b_{12}=\frac{yx\left(x^2-h_x^2\right)}{h_x^3h_y} & b_{15}=\frac{y^2\left(y^2-h_y^2\right)}{h_y^4}. \end{array} \tag{2.37}$$

As in the previous one dimensional example, the choice of basis functions results in a

reduced system

$$\begin{pmatrix} A_1 & B \\ 0 & C \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ r \end{pmatrix}. \tag{2.38}$$

In (2.38) the system

$$C\beta = r \tag{2.39}$$

is

$$\begin{aligned} \sum_{j \in \mathcal{E}_i} \beta_j (Lb_k)_j &= 0, \quad k = 10, \ldots, K \\ \sum_{j \in \mathcal{E}_i} \beta_j &= 1 \end{aligned}$$

which is solved to compute the βs.

The *HODIEX* equations must be evaluated and solved N times (at each grid point) unless the PDE coefficients are constant in which case all N solves are identical. This can become a computationally expensive operation and special care must also be taken in solving for the βs. The β system will always be *consistent* (there always exists at least one non-trivial solution), but it may be singular or nearly singular depending on the choice of evaluation points. In fact, for simple operators such as the Laplacian, the β system is always singular because several equations are identical due to the symmetry of the operator. Selection of evaluation points is discussed in the next section.

After solving for the βs, the αs are computed by solving the system

$$A_1 \alpha = -B\beta. \tag{2.40}$$

It at first appears that there are also N solves for the αs, but upon careful observation ones sees that A_1 depends only on $\{b_1, \ldots, b_9\}$ and not on L, and that from grid point to grid point only the right hand side of (2.40) changes. In solving for the αs the coefficient matrix

of the linear system is

$$A_1 = \begin{pmatrix} (b_1)_{j_1} & (b_1)_{j_2} & \cdots & (b_1)_{j_9} \\ (b_2)_{j_1} & (b_2)_{j_2} & \cdots & (b_2)_{j_9} \\ \vdots & \vdots & \ddots & \vdots \\ (b_9)_{j_1} & (b_9)_{j_2} & \cdots & (b_9)_{j_9} \end{pmatrix} \tag{2.41}$$

and the choice of basis functions makes this matrix

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & -1 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & -1 & 1 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}. \tag{2.42}$$

Once (2.42) is factored simple triangular solves yield the αs for each grid point.

After the operators L_i and I_i have been determined for $i = 1, \ldots, N$, the results are assembled into a linear system

$$Au_h = I_h s_h g \equiv g_h \tag{2.43}$$

in which the entries of the ith row of A are the αs defining L_i and the entries of the ith row of I_h are the βs defining I_i. The ith entry of g_h is the linear combination of g values obtained from the βs defining I_i.

Using the normalization equation (2.34) results in $(g_h)_i = g_i + \mathcal{O}(h)$ [90]. This is equivalent to saying that $I_h s_h$ is a perturbation of the identity operator and is the source of the term "identity expansion" in the acronym "*HODIE*".

2.3.5 An Alternative $\mathcal{O}(h^2)$ 9-Point Compact Stencil

When coefficient $b = 0$ the formulas in (2.4) result in $\mathcal{O}(h^2)$ accurate 5-point stencils. Using the techniques of the previous section for solving the *HODIEX* equations, we may obtain an alternate $\mathcal{O}(h^2)$ discretization which uses the 9-point compact stencil.

We seek a discretization on $\mathcal{P}_3$ and use the basis (for simplicity of notation again centered at the origin)

$$\begin{array}{lllll} b_1 = 1 & b_3 = \frac{x^2}{h} & b_5 = \frac{y^2}{h^2} & b_7 = \frac{x^2y}{h^3} & b_9 = \frac{x^3}{h^3} \\ b_2 = \frac{x}{h} & b_4 = \frac{y}{h} & b_6 = \frac{xy}{h^2} & b_8 = \frac{xy^2}{h^3} & b_{10} = \frac{y^3}{h^3}. \end{array} \tag{2.44}$$

There is only one evaluation point, hence $\beta_{j_1} = 1$. Applying the basis functions to the formula

$$\sum_{j \in \mathcal{D}_i} \alpha_j (b_k)_j = (Lb_k)_{j_1}, \quad k = 1, \ldots, 10 \tag{2.45}$$

at a point x_i results in a homogeneous system with 8 independent equations for the 9 stencil points (the ninth and tenth equations are duplicates of the second and third equations, respectively). A normalization equation can only involve αs since there are no other unknowns. Since we know for elliptic equations that $ac > 0$, and we would like the corner stencil points to be nonzero, we choose as a normalization equation

$$\alpha_6 + \alpha_7 + \alpha_8 + \alpha_9 = \frac{(a+c)}{3h^2}. \tag{2.46}$$

Of course other normalization equations would result in other discretizations; however, this choice results in the particular discretization we seek. The resulting linear system is

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & -1 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 1 & 1 & -1 & -1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 1 & -1 & 0 & 0 & 1 & -1 & -1 & 1 \\ 0 & 0 & 0 & 1 & -1 & 1 & 1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \\ \alpha_9 \end{pmatrix} = \begin{pmatrix} f \\ \frac{d}{h} \\ \frac{2a}{h^2} \\ \frac{e}{h} \\ \frac{2c}{h^2} \\ 0 \\ 0 \\ 0 \\ \frac{(a+c)}{3h^2} \end{pmatrix}. \quad (2.47)$$

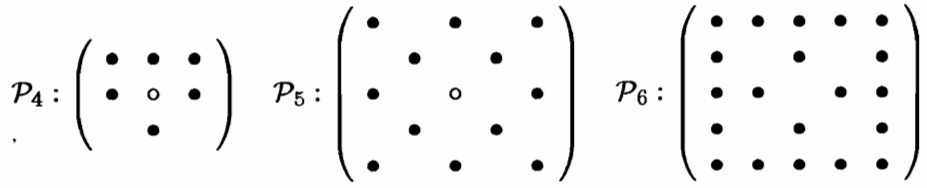
Solution of the above system yields the equation

$$\frac{1}{12h^2} \begin{array}{|c|c|c|} \hline a+c & 10c-2a-6he & a+c \\ \hline 10a-2c+6hd & -20(a+c)+12h^2f & 10a-2c-6hd \\ \hline a+c & 10c-2a+6he & a+c \\ \hline \end{array} \quad u = g + \mathcal{O}(h^2). \quad (2.48)$$

The stencil in (2.48) was derived by both Bickley [5] and Krylov [79] who also showed that the resulting linear system has properties similar to those of a linear system resulting from using formulas (2.4).

2.3.6 Selection of Evaluation Points

The selection of the set of evaluation points is a critical factor in obtaining the minimum error for a *HODIEX* discretization. Choosing $M = K - 8$ as the number of evaluation points results in a square system in which the number of unknowns equals the number of equations, but not every choice of evaluation points leads to a *HODIEX* discretization with the expected accuracy. If the choice of evaluation points makes the system underdetermined (some rows are not linearly independent), or nearly underdetermined (some rows are nearly



(2.49)

Figure 2.5: *HODIEX* evaluation points.

dependent resulting in a large condition number), then the *HODIEX* equations solve can be inaccurate and the error may increase by orders of magnitude.

For some operators (e.g., the Laplacian), symmetry allows fewer evaluation points (or equivalently some βs are zero) for a given accuracy. Higher accuracy may also be obtained by the use of special evaluation points (Gauss points) whose location depends on the operator [89]. Gauss points have the effect of two non-Gaussian points [90] resulting in the need for fewer evaluation points. In *HODIE* [91] and [14], Lynch and Boisvert take advantage of Gauss points to obtain $\mathcal{O}(h^6)$ accuracy on $\mathcal{P}_5$ when f is constant and $d_y = e_x$.

For general L, research has failed to establish algorithms for the selection of the best evaluation points. Boisvert does an analysis for the Helmholtz equation

$$u_{xx} + u_{yy} + fu = g$$

using the linear space $\mathcal{P}_5$ in which the evaluation points are the 9 stencil points and four additional points along the lines $x = y$ and $x = -y$. Boisvert shows that there is an optimal choice of the four additional evaluation points for this problem. For a complete discussion refer to Boisvert [16].

The evaluation points chosen for our numerical tests of *HODIEX* are shown in Figure 2.5. In an attempt to maximize symmetry, the central stencil point is not used for $\mathcal{P}_6$. In each case the evaluation points remain within the computational element. This means that for $\mathcal{P}_5$ and $\mathcal{P}_6$ the non-stencil evaluation points are at midpoints between grid points. This does not increase the grid size but does require additional right hand side evaluations. In the *HODIEX* code right hand sides are only evaluated once with results reused in computations at different grid points.

For Poisson's equation an example of a stencil equation using $\mathcal{P}_5$ is

$$\frac{1}{6h^2}\begin{array}{|c|c|c|}\hline 1 & 4 & 1 \\ \hline 4 & -20 & 4 \\ \hline 1 & 4 & 1 \\ \hline\end{array}\ u = \frac{1}{360}\begin{array}{|c|c|c|c|c|}\hline 1 & & 4 & & 1 \\ \hline & 48 & & 48 & \\ \hline 4 & & 148 & & 4 \\ \hline & 48 & & 48 & \\ \hline 1 & & 4 & & 1 \\ \hline\end{array}\ g + \mathcal{O}(h^6) \tag{2.50}$$

where the 48s in the right stencil are at half grid points. As shown, this stencil achieves $\mathcal{O}(h^6)$ accuracy.

2.3.7 HODIEX Properties

The *HODIEX* equations are local, dependent on each grid point, and thus the *HODIEX* discretization method requires the solution of a small linear system at each grid point. If the PDE has constant coefficients then all N systems are the same and only one set of αs and βs is required. This means that a higher order is achieved on constant coefficient PDEs at very little cost. However, even with variable coefficients, on all but the smallest systems, the *HODIEX* discretization time is small when compared with the solution time by either direct or iterative solvers, thus the higher order is computationally inexpensive.

HODIEX uses a 3×3 compact stencil, and the use of complicated nonsymmetric stencils near boundaries is not required. Evaluation points are never placed outside of a computational element and need only be evaluated once with that result used for all grid points. If only the central stencil point is used as an evaluation point then the method reduces to standard finite differences and achieves $\mathcal{O}(h^2)$ accuracy. It is the use of multiple evaluation points that gives this method high order accuracy.

In Chapter 3 we will show that *HODIEX* discretizations on $\mathcal{P}_k$, $k > 3$, with the normalization $\beta_{j_1} = 1$ are $\mathcal{O}(h)$ perturbations of the stencil in (2.48). Similarly, for sufficiently small h, the *HODIEX* linear system shares similar desirable properties. After multiplying by -1 if necessary, the coefficient matrix resulting from a *HODIEX* discretization is

- weakly diagonally dominant
- positive definite
- irreducible
- nonsingular
- an L-matrix
- of monotone type.

In addition to the above properties, other properties result from use of the compact 9-point stencil. The coefficient matrix is

- sparse with at most 9 nonzeros per row
- banded with bandwidth $n+1$
- block-tridiagonal and each block tridiagonal.

2.4 Test Problems

In order to illustrate the performance of the *HODIEX* discretization, four test problems are selected on which the true solution is known. The domain is the unit square with Dirichlet boundary conditions. Two problems have no cross derivative terms and two problems do have cross derivative terms. Each pair consists of one problem with constant PDE coefficients and one problem with variable PDE coefficients. These problems were selected from Dyksen, Houstis, and Rice [56].

Problem 1 - Variable coefficients, no cross derivative, self adjoint:

$$Lu = \left(e^{xy}u_x\right)_x + \left(e^{-xy}u_y\right)_y - \frac{u}{1+x+y}$$

with g chosen so that the true solution is

$$u(x,y) = 0.75e^{xy}\sin(\pi x)\sin(\pi y)\,.$$

Problem 2 - Constant coefficients, no cross derivative:

The Poisson problem

$$Lu = u_{xx} + u_{yy} = 6xye^{x+y}(xy + x + y - 3)$$

which has the true solution

$$u(x,y) = 3e^{x+y}\left(x - x^2\right)\left(y - y^2\right).$$

Problem 3 - Variable coefficients, cross derivative:

$$Lu = au_{xx} + bu_{xy} + cu_{yy}$$

where $a = 1 + u_x^2$, $b = -2u^2$, and $c = 1 + u_y^2$ and source term g chosen so that the true solution is

$$u(x,y) = e^{x+y}.$$

Problem 4 - Constant coefficients, cross derivative:

$$Lu = 4u_{xx} - u_{xy} + 4u_{yy}$$

where the true solution is

$$u(x,y) = (x - 3y)^2 e^{x-y}.$$

2.5 Direct Solve Performance Results

Our goal is to investigate the use of higher order discretizations with a preconditioned iterative solver, so we first want to compute the achieved order of accuracy on the set of test problems. To do this each of the test problems is run with a direct solver (banded Gaussian elimination) and the order of accuracy is computed for the various cases. The *HODIEX* discretization is computed for $\mathcal{P}_4$, $\mathcal{P}_5$, and $\mathcal{P}_6$. Results are compared with standard finite differences (equivalently discretizations on $\mathcal{P}_3$). All test were run on A DECstation 5000/125 under Ultrix, using double precision.

Tables 2.1–2.4 summarize the results. Each of the test problems are run for the five uniform grid sizes $n+1 = 8$, 16, 32, 64, and 128, chosen so that in each succeeding case the mesh size decreases by half. For each grid size, error, time, and order is presented for *HODIEX* on $\mathcal{P}_4$, $\mathcal{P}_5$, and $\mathcal{P}_6$, and standard differences on $\mathcal{P}_3$. The order r is computed by comparing the errors e_1 and e_2 for two mesh sizes h_1 and h_2 as follows:

$$\frac{Ch_1^r}{Ch_2^r} = \frac{e_1}{e_2}$$

so that

$$r = \left(\frac{\log e_1 - \log e_2}{\log h_1 - \log h_2}\right).$$

The error is measured by taking the infinity norm of the error over the grid points.

Figures 2.6–2.9 are *log* − *log* plots of *Time* versus *Error*. The labels $\mathcal{P}_4$ through $\mathcal{P}_6$ indicate the polynomial space on which the *HODIEX* equations are required to be exact. $\mathcal{P}_3$ corresponds to standard finite differences. On all test problems, standard finite differences on $\mathcal{P}_3$ achieves $\mathcal{O}(h^2)$ accuracy as expected.

The theory (to be presented in Chapter 3) indicates that when the *HODIEX* discretization exists on $\mathcal{P}_k$ we can expect $\mathcal{O}(h^{k-1})$ accuracy. On the problems with a cross derivative the results agree with the theory on $\mathcal{P}_4$ and $\mathcal{O}(h^3)$ accuracy is achieved. The problems with no cross derivative achieved $\mathcal{O}(h^4)$ accuracy on $\mathcal{P}_4$. It is not uncommon for *HODIE* methods on $\mathcal{P}_k$ to achieve $\mathcal{O}(h^k)$ accuracy [90].

Going to $\mathcal{P}_5$ did not significantly improve the accuracy for problems with no cross derivative, but did increase the accuracy for problems with cross derivative to $\mathcal{O}(h^4)$. Thus, each problem achieved $\mathcal{O}(h^4)$ accuracy on $\mathcal{P}_5$ as expected; however, since some problems perform so well on $\mathcal{P}_4$, the extra time spent on $\mathcal{P}_5$ is not worthwhile.

On $\mathcal{P}_6$ the results vary from failure of the *HODIEX* discretization to exist, to no improvement over $\mathcal{P}_5$, and to the approximation being nearly exact on some higher dimensional space. The failure of the *HODIEX* discretization to exist on problem 4 will be characterized in terms of the null space of the operator in Chapter 3. On problems 1 and 3 the *HODIEX* discretization exists, but there is no improvement in the order of accuracy over

$\mathcal{P}_5$. This could indicate a poor selection of evaluation points. On the other hand, the results for problem 2 indicates a good choice of evaluation points. The $\mathcal{O}(h^8)$ accuracy achieved indicates that the choice of evaluation points include Gauss type points and that actually the approximation is nearly exact on some higher dimensional space such as $\mathcal{P}_8$ or $\mathcal{P}_9$.

Figures 2.6–2.9 indicate the effectiveness of *HODIEX* over standard finite differences. For a given error, *HODIEX* is much faster than the $\mathcal{O}(h^2)$ method. Similarly, for a given time, *HODIEX* produces a smaller error than standard finite differences. In all cases *HODIEX* produces at least $\mathcal{O}(h^3)$ accuracy ($\mathcal{O}(h^4)$ with no cross derivative term). Thus our goal is achieved and we will proceed to investigate the use of higher order discretizations with a preconditioned iterative solver.

Table 2.1: Problem 1. Log of Time vs. Log of Error.

	HODIEX $\mathcal{P}_4$			*HODIEX* $\mathcal{P}_5$		
n+1	Error	Time	Order	Error	Time	Order
8	$6.1E-5$	.01		$1.6E-5$	.01	
16	$4.9E-6$	.06	3.63	$1.2E-6$	.15	3.74
32	$3.4E-7$	.30	3.85	$8.5E-8$	.72	3.82
64	$2.2E-8$	2.11	3.95	$5.6E-9$	3.87	3.92
128	$1.4E-9$	25.52	3.97	$3.6E-10$	32.92	3.96
	HODIEX $\mathcal{P}_6$			Standard Differences $\mathcal{P}_3$		
n+1	Error	Time	Order	Error	Time	Order
8	$6.9E-4$	.08		$8.6E-3$	.01	
16	$6.7E-5$	.38	3.36	$2.5E-3$	.03	1.78
32	$4.7E-6$	1.65	3.83	$6.5E-4$	.11	1.94
64	$4.8E-7$	7.72	3.30	$1.7E-4$	1.42	1.94
128	$6.5E-8$	48.53	2.89	$4.3E-5$	20.20	1.98

Table 2.2: Problem 2. Log of Time vs. Log of Error.

	HODIEX $\mathcal{P}_4$			*HODIEX* $\mathcal{P}_5$		
n+1	Error	Time	Order	Error	Time	Order
8	$1.4E-5$	.01		$1.0E-6$	.01	
16	$1.1E-6$	.01	3.67	$6.4E-8$	.02	3.97
32	$7.9E-8$	.08	3.80	$3.4E-9$	.09	4.23
64	$5.2E-9$	1.23	3.93	$2.9E-10$	1.29	3.55
128	$3.3E-10$	22.69	3.98	$5.1E-11$	23.10	2.51
	HODIEX $\mathcal{P}_6$			Standard Differences $\mathcal{P}_3$		
n+1	Error	Time	Order	Error	Time	Order
8	$8.6E-6$	.01		$3.5E-3$	.01	
16	$3.4E-7$	.01	4.66	$9.7E-4$	.01	1.85
32	$1.3E-9$	.10	8.03	$2.6E-4$	.08	1.90
64	$5.5E-12$	1.31	7.88	$6.7E-5$	1.26	1.96
128	$2.8E-14$	23.29	7.61	$1.7E-5$	21.28	1.98

Table 2.3: Problem 3. Log of Time vs. Log of Error.

	HODIEX $\mathcal{P}_4$			*HODIEX* $\mathcal{P}_5$		
n+1	Error	Time	Order	Error	Time	Order
8	$8.5E-6$	.01		$9.6E-8$	.02	
16	$1.5E-6$	.03	2.50	$7.6E-9$	.10	3.66
32	$2.2E-7$	.21	2.77	$5.3E-10$	.50	3.84
64	$3.0E-8$	1.79	2.87	$3.5E-11$	3.04	3.92
128	$3.9E-9$	24.07	2.94	$2.9E-12$	28.94	3.69
	HODIEX $\mathcal{P}_6$			Standard Differences $\mathcal{P}_3$		
n+1	Error	Time	Order	Error	Time	Order
8	$9.4E-9$	.06		$1.4E-3$	.01	
16	$8.6E-10$	.27	3.45	$3.9E-4$	.01	1.84
32	$7.4E-11$	1.25	3.54	$1.1E-4$	.10	1.83
64	$5.5E-12$	5.99	3.75	$2.8E-5$	1.35	1.97
128	$4.0E-13$	30.95	3.78	$7.0E-6$	19.54	2.00

Table 2.4: Problem 4. Log of Time vs. Log of Error.

	HODIEX $\mathcal{P}_4$			*HODIEX* $\mathcal{P}_5$		
n+1	Error	Time	Order	Error	Time	Order
8	$1.0E-5$	.01		$5.2E-6$	.01	
16	$1.3E-6$	.01	2.94	$4.1E-7$	.01	3.66
32	$1.5E-7$	.09	3.12	$2.9E-8$	.10	3.82
64	$1.9E-8$	1.20	2.98	$1.9E-9$	1.31	3.93
128	$2.3E-9$	21.67	3.05	$1.2E-10$	22.14	3.98
	HODIEX $\mathcal{P}_6$			Standard Differences $\mathcal{P}_3$		
n+1	Error	Time	Order	Error	Time	Order
8				$1.1E-2$	.01	
16				$3.2E-3$	.01	1.78
32				$8.5E-4$	.08	1.91
64				$2.2E-4$	1.28	1.95
128				$5.5R-5$	19.21	2.00

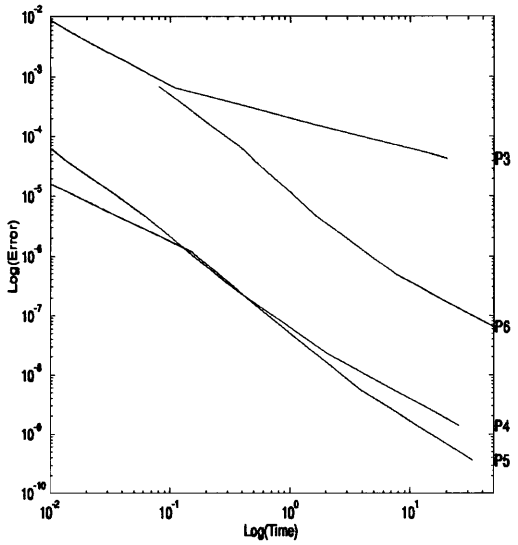


Figure 2.6: Problem 1. $Lu = \left(e^{xy}u_x\right)_x + \left(e^{-xy}u_y\right)_y - \frac{u}{1+x+y}$.

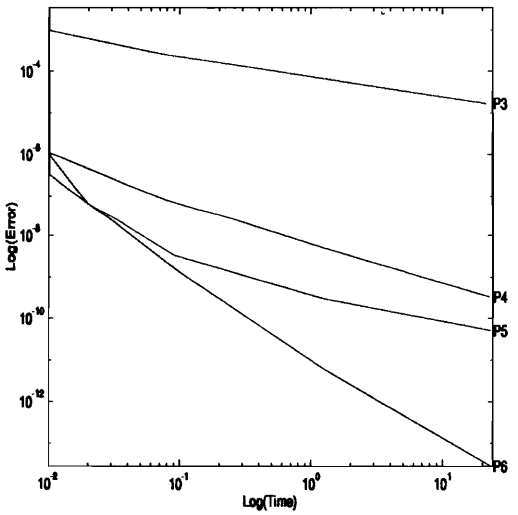


Figure 2.7: Problem 2. $Lu = u_{xx} + u_{yy}$.

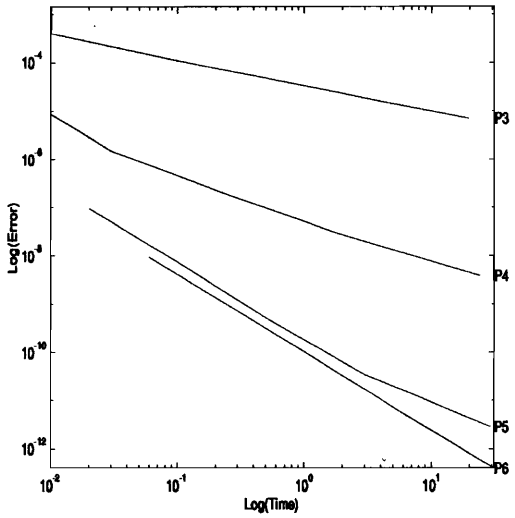


Figure 2.8: Problem 3. $Lu = au_{xx} + bu_{xy} + cu_{yy}$.

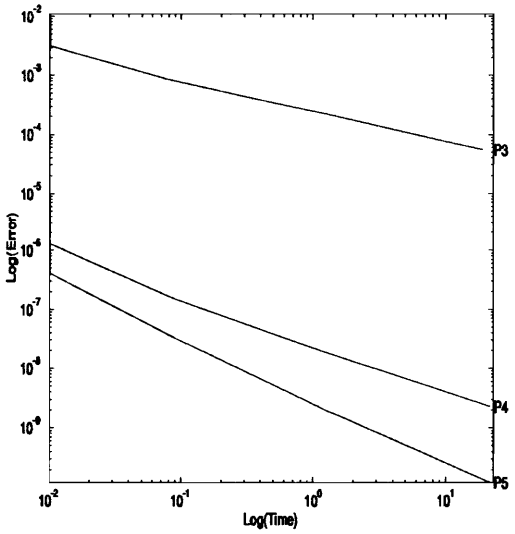


Figure 2.9: Problem 4. $Lu = 4u_{xx} - u_{xy} + 4u_{yy}$.

Chapter 3

HODIEX THEORY

In this chapter we show that the linear system resulting from a *HODIEX* discretization is closely related to the linear system resulting from standard finite differences and has similar desirable properties. We consider the question of existence of the *HODIEX* discretization and give conditions under which a *HODIEX* discretization exact on certain polynomial spaces fails to exist. The maximum achievable order of approximation (accuracy) of the *HODIEX* discretization is shown to depend both on the coefficients of the PDE and the linear space used. Finally we show that a non-trivial *HODIEX* discretization has truncation error and discretization error of the same order.

3.1 The HODIEX Coefficients

In this section we show that the α coefficients of a *HODIEX* discretization are $\mathcal{O}(h)$ (or if $d = e = 0$ then $\mathcal{O}(h^2)$) perturbations of the classical Krylov 9 point finite difference coefficients shown in Equation (3.5), and thus, for sufficiently small h, the coefficient matrix of the linear system has properties similar to those of the coefficient matrix resulting from a discretization using (3.5) (Bickley [5] and Krylov [79]).

It is shown by Young and Gregory [122] that there always exists a real change of variables

$$\hat{x} = \hat{x}(x, y) \tag{3.1}$$

$$\hat{y} = \hat{y}(x, y) \tag{3.2}$$

which results in elimination of the cross derivative term in (2.1) and does not change the classification (elliptic, hyperbolic, or parabolic) of the PDE. He shows that $\hat{x}$ and $\hat{y}$ must

satisfy

$$\frac{\hat{x}_y}{\hat{x}_x} = \sqrt{\frac{a(x,y)}{c(x,y)}} \tag{3.3}$$

and

$$\frac{\hat{y}_y}{\hat{y}_x} = -\sqrt{\frac{a(x,y)}{c(x,y)}} \tag{3.4}$$

and gives a numerical quadrature technique that yields $\hat{x}$ and $\hat{y}$ for given x and y. See Young and Gregory [122] for details. Without loss of generality we assume that $b = 0$ in Equation (2.1). We also assume that (2.1) is well posed, so there exists a unique solution $u = L^{-1}g$ which depends continuously on the input data.

Krylov's [79] 9-point compact central finite difference stencil for Equation (2.1)

$$\frac{1}{12h^2}\begin{array}{|c|c|c|}\hline a+c & 10c-2a-6he & a+c \\ \hline 10a-2c+6hd & -20(a+c)+12h^2f & 10a-2c-6hd \\ \hline a+c & 10c-2a+6he & a+c \\ \hline\end{array} \quad u = g + \mathcal{O}(h^2). \tag{3.5}$$

leads to discretizations that have an order of accuracy $\mathcal{O}(h^2)$ and are exact on $\mathcal{P}_3$.

We will use the following notation in describing the α and β coefficients of various operator discretizations:

Discretization	Alpha	Beta	Operator and Coefficients
HODIEX	$\hat{\alpha}$	$\hat{\beta}$	$L_2u = au_{xx} + cu_{yy}$, a and c constant
HODIEX	$\bar{\alpha}$	$\bar{\beta}$	$L_1u = au_{xx} + cu_{yy} + du_x + eu_y + fu$, a and c constant
HODIEX	α	β	$Lu = au_{xx} + cu_{yy} + du_x + eu_y + fu$, a–f variable
HODIEX	$\breve{\alpha}$	$\breve{\beta}$	$L_0u = au_{xx}+cu_{yy}+du_x+eu_y+fu$, a–f piecewise constant
(3.5)	$\acute{\alpha}$	$\acute{\beta}_1 = 1$	$Lu = au_{xx} + cu_{yy} + du_x + eu_y + fu$, a–f variable

We will show that *HODIEX* discretizations exact on spaces which contain $\mathcal{P}_4$ generate α stencils which are $\mathcal{O}(h)$ *relative* perturbations of the stencil in Equation (3.5). For example, when we say that a is a $\mathcal{O}(h^k)$ relative perturbation of b we mean $a = b\left(1 + \mathcal{O}(h^k)\right)$. The following theorems are from Boisvert [16], Lynch and Rice [90], Birkhoff and Gulati [7],

and Young [122]. They are presented here to summarize the theory supporting *HODIEX*. The proofs assume the basis for $\mathcal{P}_k$ given in the previous chapter in which (for notational convenience) each function has been translated to $(0,0)$.

Theorem 3.1 *Consider a HODIEX discretization exact on $\mathcal{P}_k$ for any $k \geq 4$ of*

$$L_2 u = a u_{xx} + c u_{yy}$$

where a and c are constant. Assume that $\hat{\alpha}$ and $\hat{\beta}$ are scaled to $\hat{\beta}_1 = 1$, then

$$\|\hat{\alpha}\| = \mathcal{O}(h^{-2}) \tag{3.6}$$

$$\|\hat{\beta}\| = \mathcal{O}(1). \tag{3.7}$$

Proof: The *HODIEX* equations have the $K \times K$ form (no normalization equation)

$$\begin{pmatrix} A_1 & B \\ 0 & C \end{pmatrix} \begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \end{pmatrix} = \begin{pmatrix} q \\ r \end{pmatrix}$$

where q and r are the result of bringing the terms involving $\hat{\beta}_1$ to the right hand side. Each element of B, C, q, and r has the form

$$\begin{aligned} (Lp_k)_j &= \left(L \frac{x^s y^t}{h^s h^t} \right)_j \\ &= \left(as(s-1)\frac{x^{s-2}y^t}{h^s h^t} + ct(t-1)\frac{x^s y^{t-2}}{h^s h^t} \right)_j, \end{aligned}$$

where s, $t = 0, 1, \ldots$, and $s + t \leq k$. Since $-h \leq x, y \leq +h$ then the *HODIEX* equations may be rewritten in the form

$$\begin{pmatrix} A_1 & B_1 h^{-2} \\ 0 & C_1 h^{-2} \end{pmatrix} \begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \end{pmatrix} = \begin{pmatrix} q_1 h^{-2} \\ r_1 h^{-2} \end{pmatrix}$$

in which A_1, B_1, C_1, q_1, and r_1 are constant and independent of h. In particular, in the equation

$$C_1 h^{-2} \hat{\beta} = r_1 h^{-2}$$

both C_1 and r_1 are constants independent of h and thus it is clear that $\hat{\beta} = \mathcal{O}(1)$.

In the equation

$$A_1\hat{\alpha} + Bh^{-2}\hat{\beta} = q_1 h^{-2}$$

all constants are similarly independent of h and it is clear that $\hat{\alpha} = \mathcal{O}(h^{-2})$. •

Theorem 3.2 *Consider HODIEX discretizations exact on $\mathcal{P}_k$ for any $k \geq 4$ of*

$$L_1 u = au_{xx} + cu_{yy} + du_x + eu_y + fu \tag{3.8}$$

$$L_2 u = au_{xx} + cu_{yy} \tag{3.9}$$

where a, c, d, e, and f are constant. Then

$$\bar{\beta} = \hat{\beta} + \mathcal{O}(h^r) \tag{3.10}$$

$$\bar{\alpha} = \hat{\alpha}\,(1 + \mathcal{O}(h^r)) \tag{3.11}$$

where $r = 1$ unless $d = e = 0$, in which case $r = 2$.

Proof: Consider (3.8). The *HODIEX* equations can be written as the $K \times (K+1)$ system (no normalization equation)

$$\begin{pmatrix} A_1 & B \\ 0 & C \end{pmatrix} \begin{pmatrix} \bar{\alpha} \\ \bar{\beta} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Each element of B and C has the form

$$\begin{aligned} (Lp_k)_j &= \left(L\frac{x^s y^t}{h^s h^t} \right)_j \\ &= \left(as(s-1)\frac{x^{s-2}y^t}{h^s h^t} + ct(t-1)\frac{x^s y^{t-2}}{h^s h^t} + ds\frac{x^{s-1}y^t}{h^s h^t} + et\frac{x^s y^{t-1}}{h^s h^t} + f\frac{x^s y^t}{h^s h^t} \right)_j \end{aligned}$$

and since $-h \leq x, y \leq +h$ the *HODIEX* equations for (3.8) may be rewritten in the form

$$\begin{pmatrix} A_1 & B_0 + B_1 h^{-1} + B_2 h^{-2} \\ 0 & C_0 + C_1 h^{-1} + C_2 h^{-2} \end{pmatrix} \begin{pmatrix} \bar{\alpha} \\ \bar{\beta} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{3.12}$$

Similarly, the *HODIEX* equations for (3.9), which has only second derivatives, can be written in the form

$$\begin{pmatrix} A_1 & B_2h^{-2} \\ 0 & C_2h^{-2} \end{pmatrix} \begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{3.13}$$

where the B_j and C_j are independent of h. B_0 and C_0 come from the u term in (3.8), B_1 and C_1 come from the first derivative terms, and B_2 and C_2 come from the second derivative terms. Subtracting (3.13) from (3.12) yields the two equations

$$A_1(\bar{\alpha} - \hat{\alpha}) + B_0\bar{\beta} + B_1h^{-1}\bar{\beta} + B_2h^{-2}(\bar{\beta} - \hat{\beta}) = 0, \tag{3.14}$$

$$C_0\bar{\beta} + C_1h^{-1}\bar{\beta} + C_2h^{-2}(\bar{\beta} - \hat{\beta}) = 0. \tag{3.15}$$

From (3.15) it is clear that $\bar{\beta} - \hat{\beta} = \mathcal{O}(h)$. From (3.14) it is clear that $\bar{\alpha} - \hat{\alpha} = \mathcal{O}(h^{-1})$. Since $\hat{\alpha} = \mathcal{O}(h^{-2})$ it follows that

$$\begin{aligned} \bar{\alpha} &= \hat{\alpha} + \mathcal{O}(h^{-1}) \\ &= \hat{\alpha} + \hat{\alpha}\mathcal{O}(h) \\ &= \hat{\alpha}\,(1 + \mathcal{O}(h))\,. \end{aligned}$$

If $d = e = 0$ then $B_1 = C_1 = 0$. From (3.15) it is clear that $\bar{\beta} - \hat{\beta} = \mathcal{O}(h^2)$. From (3.14) it is clear that $\bar{\alpha} - \hat{\alpha} = \mathcal{O}(h^0)$. Since $\hat{\alpha} = \mathcal{O}(h^{-2})$ it follows that

$$\begin{aligned} \bar{\alpha} &= \hat{\alpha} + \mathcal{O}(h^0) \\ &= \hat{\alpha} + \hat{\alpha}\mathcal{O}(h^2) \\ &= \hat{\alpha}\left(1 + \mathcal{O}(h^2)\right). \end{aligned}$$

•

Theorem 3.3 *Consider a HODIEX discretization exact on $\mathcal{P}_k$ for any $k \geq 4$ of*

$$L_2 u = a u_{xx} + c u_{yy} \tag{3.16}$$

and the discretization in Equation (3.5) where a and c are constant and $d = e = f = 0$. Then $\hat{\alpha} = \acute{\alpha}$.

Proof: The proof is mostly computational. The details are given by Boisvert [16]. •

Theorem 3.4 *Consider a HODIEX discretization exact on $\mathcal{P}_k$ for any $k \geq 4$ of*

$$L_1 u = a u_{xx} + c u_{yy} + d u_x + e u_y + f u \tag{3.17}$$

and the discretization in Equation (3.5) where a, c, d, e, and f are constant. Then $\bar{\alpha} = \acute{\alpha}(1 + \mathcal{O}(h))$. If $d = e = 0$ then $\bar{\alpha} = \acute{\alpha}(1 + \mathcal{O}(h^2))$.

Proof: We apply the previous two theorems. By Theorem 3.3 $\hat{\alpha} = \acute{\alpha}$ and by Theorem 3.2 $\bar{\alpha} = \hat{\alpha}\,(1 + \mathcal{O}(h))$; hence $\bar{\alpha} = \acute{\alpha}\,(1 + \mathcal{O}(h))$. If $d = e = 0$ then $\bar{\alpha} = \hat{\alpha}\,(1 + \mathcal{O}(h^2))$; hence $\bar{\alpha} = \acute{\alpha}\,(1 + \mathcal{O}(h^2))$. •

The preceding theorems have dealt with constant PDE coefficients. These results are now extended to the case of variable coefficients.

Theorem 3.5 *Consider a HODIEX discretization exact on $\mathcal{P}_k$ for any $k \geq 4$ of*

$$Lu = au_{xx} + cu_{yy} + du_x + eu_y + fu \tag{3.18}$$

Let L_0 be the piecewise constant coefficient operator obtained from L by evaluating each coefficient function only at the central stencil point. Then the coefficients $\acute{\alpha}$ at grid point p_i satisfying the equations

$$\sum_{j \in \mathcal{D}_i} \acute{\alpha}_j (b_k)_j - \sum_{j \in \mathcal{E}_i} \acute{\beta}_j (L_0 b_k)_j = 0, \quad k = 1, \ldots, K$$

also satisfy $\alpha = \acute{\alpha}\,(1 + \mathcal{O}(h))$. If $d = e = 0$ then $\alpha = \acute{\alpha}\,(1 + \mathcal{O}(h^2))$

Proof: Consider (3.18). In a manner similar to the proof of Theorem 3.2 the *HODIEX* equations of (3.18) can be written as

$$\begin{pmatrix} A_1 & B \\ 0 & C \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{3.19}$$

B and C can be written as

$$B = B_0 + B_1 h^{-1} + B_2 h^{-2}$$

$$C = C_0 + C_1 h^{-1} + C_2 h^{-2}$$

where each B_j and C_j are independent of h. Since the coefficients are $\mathcal{C}^2(\mathcal{D}$ functions, at each grid point $(x_0, y_0) \in \mathcal{D}$ we can expand the coefficient functions using a Taylor series in the form

$$a(x, y) = a(x_0, y_0) + \mathcal{O}(h).$$

Since the elements of each B_j and C_j are linear combinations of values of the coefficient functions, they may be expanded in the form

$$B_j = B_{j0} + B_{j1} h$$

and

$$C_j = C_{j0} + C_{j1} h$$

where B_{jk} and C_{jk} are $\mathcal{O}(1)$. Thus B and C can be written

$$\begin{aligned} B &= (B_{00} + B_{10}h^{-1} + B_{20}h^{-2}) + (B_{01} + B_{11}h^{-1} + B_{21}h^{-2})h \\ &\equiv \hat{B}_0 + \hat{B}_1 h \\ C &= (C_{00} + C_{10}h^{-1} + C_{20}h^{-2}) + (C_{01} + C_{11}h^{-1} + C_{21}h^{-2})h \\ &\equiv \hat{C}_0 + \hat{C}_1 h \end{aligned}$$

where $\hat{B}_0$, $\hat{B}_1$, $\hat{C}_0$, and $\hat{C}_1$ are $\mathcal{O}(h^{-2})$. We can then write the *HODIEX* equations for L as

$$\begin{pmatrix} A_1 & \hat{B}_0 + \hat{B}_1 h \\ 0 & \hat{C}_0 + \hat{C}_1 h \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \tag{3.20}$$

and for L_0

$$\begin{pmatrix} A_1 & \hat{B}_0 \\ 0 & \hat{C}_0 \end{pmatrix} \begin{pmatrix} \acute{\alpha} \\ \acute{\beta} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{3.21}$$

This representation for the discretization of L_0 follows from holding the coefficients constant at each grid point and thus independent of h. Subtracting (3.21) from (3.20) yields the two equations

$$A_1(\alpha - \acute{\alpha}) + \hat{B}_0(\beta - \acute{\beta}) + \hat{B}_1 \beta h = 0 \tag{3.22}$$

$$\hat{C}_0(\beta - \acute{\beta}) + \hat{C}_1 \beta h = 0. \tag{3.23}$$

From Equation (3.23) it is clear that $\beta - \acute{\beta} = \mathcal{O}(h)$ or $\beta = \acute{\beta} + \mathcal{O}(h)$. Substituting this into Equation (3.22) gives

$$\mathcal{O}(1)(\alpha - \acute{\alpha}) + \mathcal{O}(h^{-2})\mathcal{O}(h) + \mathcal{O}(h^{-2})\mathcal{O}(1)h = 0$$

which gives $\alpha - \acute{\alpha} = \mathcal{O}(h^{-1})$ and thus $\alpha = \acute{\alpha}(1 + \mathcal{O}(h))$. •

The main result then follows.

Theorem 3.6 *Consider a HODIEX discretization exact on $\mathcal{P}_k$ for any $k \geq 4$ of*

$$Lu = au_{xx} + cu_{yy} + du_x + eu_y + fu \tag{3.24}$$

and the discretization in Equation (3.5) where a, c, d, e, and f are variable. Then $\alpha = \acute{\alpha}(1 + \mathcal{O}(h))$. If $d = e = 0$ then $\alpha = \acute{\alpha}(1 + \mathcal{O}(h^2))$.

Proof: This result follows directly from the previous theorem. At each grid point the *HODIEX* discretization is an $\mathcal{O}(h)$ (or $\mathcal{O}(h^2)$ if $d = e = 0$) perturbation of the discretization obtained by holding the PDE coefficients constant with their value determined by the central grid point. But each discretization of these constant coefficient operators is itself an $\mathcal{O}(h)$ (or $\mathcal{O}(h^2)$ if $d = e = 0$) perturbation of the second order 9-point discretization (3.5).

•

Although the above results are for $h_x = h_y = h$, similar results may be obtained for $h_x \neq h_y$.

3.2 Existence of the HODIEX Discretization

In this section we present a series of results which characterize when a nontrivial *HODIEX* approximation does or does not exist. If the only solution to the *HODIEX* equations is the trivial solution, that is

$$\alpha_{j_k} = 0, \quad k = 1, \ldots, 9,$$

then no *useful HODIEX* discretization exists, or equivalently, we say that the *HODIEX* discretization *does not exist.* Although we generally assume that the M evaluation points are chosen so as to assure the system of *HODIEX* equations is consistent (at least one non-trivial solution), there are conditions under which no non-trivial solution exists. The conditions which characterize such a failure are based on the null space $\mathcal{N}_L$ of L.

Definition 3.1 *The null space of the differential operator L is the set $\mathcal{N}_L$ of all functions $v \in \mathcal{C}^2$ such that $Lv = 0$.*

Consider the *HODIEX* discretization which we would like to be exact on a linear space $\mathcal{P}$ with basis functions $\{b_1, b_2, \ldots, b_K\}$. Let $\mathcal{N} = \mathcal{P} \cap \mathcal{N}_L$. Note that $\mathcal{N}$ may be the empty set. Since $\mathcal{N} \subset \mathcal{P}$ then a subset of the basis functions for $\mathcal{P}$ will form a basis for $\mathcal{N}$. We will denote this basis for $\mathcal{N}$ by $\{n_m \mid m = 1, \ldots, dim(\mathcal{N}) \leq K\}$.

Definition 3.2 *Let S be a set of $q > 0$ points*

$$S = \{s_m \mid s_m \in \mathcal{R}^2, \ m = 1, \ldots, q\} \tag{3.25}$$

and T be a set of $r \geq q$ functions

$$T = \{t_m \mid t_m : \mathcal{R}^2 \rightarrow \mathcal{R}, \ m = 1, \ldots, r\}. \tag{3.26}$$

Let $\{t_{j_1}, \ldots, t_{j_q}\}$ be any set of q distinct elements of T. If the matrix

$$\begin{pmatrix} t_{j_1}(s_1) & \ldots & t_{j_1}(s_q) \\ \vdots & \ddots & \vdots \\ t_{j_q}(s_1) & \ldots & t_{j_q}(s_q) \end{pmatrix} \tag{3.27}$$

is nonsingular for every choice of j_1 to j_q, we say that T is **linearly independent with respect to** S.

Theorem 3.7 *Given a linear elliptic operator L and a computational element $\mathcal{D}_i$, there is no non-trivial HODIEX discretization exact on $\mathcal{P}$ if $\{n_m \mid m = 1, \ldots, dim(\mathcal{N})\}$ is linearly independent with respect to $\mathcal{D}_i$.*

Proof: By the definition of linear independence $dim(\mathcal{N}) \geq 9$. Since $\mathcal{N} \subset \mathcal{P}$ we can assume a basis for $\mathcal{P}$ is

$$n_1, \ldots, n_9, b_{10}, \ldots, b_K$$

where $n_1, \ldots, n_9$ are nine of the basis functions of $\mathcal{N}$. Since $\mathcal{N} \subset \mathcal{N}_L$ then $Ln_k = 0$ for $k = 1, \ldots, 9$, and so the *HODIEX* equations (without the normalization equation) have the form

$$\begin{pmatrix} A_1 & 0 \\ A_2 & C \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{3.28}$$

Thus

$$A_1\alpha = 0,$$

and since A_1 is non-singular by the linear independence of $\mathcal{N}$ with respect to $\mathcal{D}_i$, we have $\alpha = 0$. •

The following theorem follows directly from Theorem 3.7. It shows that if $\mathcal{P}_m \bigcap \mathcal{N}_L$ is linearly independent with respect to $\mathcal{D}_i$ then there is no useful *HODIEX* discretization exact on $\mathcal{P}_n$ for $n \geq m$.

Theorem 3.8 *If $\mathcal{P} \bigcap \mathcal{N}_L$ is linearly independent with respect to $\mathcal{D}_i$ and $\mathcal{P} \subset \mathcal{Q}$, then there exists no non-trivial HODIEX discretization exact on $\mathcal{Q}$.*

Proof: $\mathcal{N} = \mathcal{P} \cap \mathcal{N}_L \subset \mathcal{Q} \cap \mathcal{N}_L$ so the basis elements of $\mathcal{Q}$ can be reordered as

$$n_1, \ldots n_9, q_{10}, \ldots, q_{dim(\mathcal{Q})}$$

and the result follows in the same manner as in the previous proof. •

The null space is perhaps more easily related to *HODIEX* by the following theorem.

Theorem 3.9 *If there exist basis functions $\{n_1, \ldots, n_9\}$ for $\mathcal{N} = \mathcal{P} \bigcap \mathcal{N}_L$ such that A_1 in Equation (2.38) is non-singular, then there is no non-trivial solution to the HODIEX equations exact on $\mathcal{P}$.*

Proof: The proof is essentially the same as for Theorem 3.7, only we do not need to renumber the basis. The *HODIEX* equations have the form

$$\begin{pmatrix} A_1 & 0 \\ A_2 & C \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \tag{3.29}$$

thus

$$A_1\alpha = 0$$

and since A_1 is non-singular then $\alpha = 0$. •

The following theorem is used to prove Theorem 3.11.

Theorem 3.10 *The coefficient α_{j_1} of a non-trivial HODIEX discretization exact on $\mathcal{P}_k$, $k \geq 3$ of*

$$Lu = au_{xx} + cu_{yy} + du_x + eu_y + fu \tag{3.30}$$

is non-zero for sufficiently small h.

Proof: The proof follows directly from Theorem 3.6 where we showed that the coefficient matrix is an $\mathcal{O}(h)$ perturbation of the coefficient matrix generated by (3.5). This implies that α_{j_1} is an $\mathcal{O}(h)$ perturbation of the center term $-20(a+c) + 12h^2 f$ in (3.5). Since a and c have the same sign, are nonzero, and f is uniformly bounded, this term is nonzero for small enough h. •

The following theorem characterizes, in another way, a situation where no non-trivial *HODIEX* discretization exists.

Theorem 3.11 *If there exists a function $p \in \mathcal{P} \bigcap \mathcal{N}_L$ such that $p = 0$ at each stencil point except the central stencil point, then no useful HODIEX discretization exists exact on $\mathcal{P}$.*

Proof: At grid point i we have

$$\begin{aligned} 0 &= L_i r_h p - I_i s_h L p \quad \text{since } p \in \mathcal{P} \\ &= L_i r_h p \quad \text{since } p \in \mathcal{N}_L \\ &= \sum_{j \in \mathcal{D}_i} \alpha_j(p)_j \quad \text{by definition} \\ &= \alpha_{j_1} p_i \end{aligned}$$

and since $p_i \neq 0$ then $\alpha_{j_1} = 0$. However, Theorem 3.10 shows that $\alpha_{j_1} \neq 0$ for a useful *HODIEX* discretization. •

A theorem by Birkhoff and Gulati [7] is easy to prove using the above result.

Theorem 3.12 *There is no useful HODIEX discretization of the Laplacian exact on* $\mathcal{P}_8$.

Proof: The function

$$p = x^8 - 28x^6y^2 + 70x^4y^4 - 28x^2y^6 + y^8 + 3h^4x^4 - 6x^2y^2 + y^4 - 4h^8$$

is harmonic and vanishes at each stencil point except the central stencil point. •

Lynch and Rice [90] prove the following result in which the coefficients of the PDE determine the achievable order.

Theorem 3.13

(i) *If* $Lu = u_{xx} + u_{yy}$ *then a non-trivial HODIEX discretization exists exact on* $\mathcal{P}_7$ *with* $\mathcal{O}(h^6)$ *error, but not exact on* $\mathcal{P}_8$.

(ii) *If* $Lu = u_{xx} + cu_{yy}$ *or* $Lu = u_{xx} + 2bu_{xy} + u_{yy}$ *with* $b \neq 0$ *and* $0 \neq c \neq 1$, *then a non-trivial HODIEX discretization exists exact on* $\mathcal{P}_5$ *but not exact on* $\mathcal{P}_6$.

(iii) *If* $Lu = u_{xx} + 2bu_{xy} + cu_{yy}$ *with* $b \neq 0$ *and* $0 \neq c \neq 1$, *then a non-trivial HODIEX discretization exists exact on* $\mathcal{P}_3$ *but not exact on* $\mathcal{P}_4$.

3.3 Convergence of the HODIEX Discretization

Due to the use of multiple right hand side evaluation points, the *HODIEX* truncation and discretization errors are defined slightly differently for *HODIEX* than for standard finite difference discretizations. The discrete linear system resulting from a *HODIEX* discretization may be written

$$Au_h = g_h = I_h s_h g$$

where the entries in each row of A correspond to αs and the entries in each row of I_h correspond to βs. Recall that A is a weakly diagonally dominant L-matrix of monotone

type and thus A is positive definite and A^{-1} exists.

The discretization error measures the difference over the grid points between the true solution u and the approximate solution u_h. Thus at grid point i

$$\begin{aligned}(r_h u)_i - (u_h)_i &= (r_h L^{-1} g)_i - (A^{-1} I_h s_h g)_i \\ &= \left((r_h L^{-1} - A^{-1} I_h s_h) g\right)_i \\ &= (e_h g)_i.\end{aligned} \tag{3.31}$$

We thus define the *discretization error* of a *HODIEX* discretization by

$$e_h g = r_h L^{-1} g - A^{-1} I_h s_h g \tag{3.32}$$

and we say that a method is *convergent* if the discretization error tends to 0 for for each choice of g.

The *truncation error* measures the difference over the grid points in approximating the continuous differential operator L by its discrete approximation A and is closely related to the discretization error. The truncation error of a *HODIEX* discretization is defined by

$$t_h u = A r_h u - I_h s_h L u. \tag{3.33}$$

We say that a method is *consistent* if the order of the truncation error is at least one, that is if

$$\|t_h u\| \leq \mathcal{O}(h). \tag{3.34}$$

Stability is a property of the discrete system. A method is said to be *stable* if there exists a constant B *independent* of h such that

$$\|A^{-1}\| \leq B. \tag{3.35}$$

The following theorem shows the relationship between discretization error, truncation error, and stability.

Theorem 3.14

$$\|e_h g\| \leq \|A^{-1}\| \, \|t_h u\| \tag{3.36}$$

Proof: We have

$$\begin{aligned} t_h u &= A r_h u - I_h s_h L u \\ &= A r_h u - I_h s_h g \\ &= A r_h u - A u_h \\ &= A(r_h u - u_h) \\ &= A e_h g. \end{aligned} \tag{3.37}$$

Thus

$$e_h g = A^{-1} t_h u \tag{3.38}$$

or

$$\|e_h g\| = \|A^{-1} t_h u\| \tag{3.39}$$

$$\leq \|A^{-1}\| \|t_h u\|. \tag{3.40}$$

•

Thus if the discretization is stable and consistent it will be convergent. We assume stability, thus for the coefficient matrix A of the discrete linear system generated by any *HODIEX* discretization for any h then $\|A^{-1}\|$ is bounded independent of h. The following theorem shows that the *HODIEX* discretization is consistent since the order of the truncation error is at least one.

Theorem 3.15 *If $u \in C^{k+1}(D)$ then the HODIEX discretization exact on $\mathcal{P}_k$ has truncation error $\mathcal{O}(h^{k-1})$.*

Proof: Since $u \in \mathcal{C}^{k+1}(D)$ we can expand u at each grid point by a Taylor series to get $u = p + q$ where $p \in \mathcal{P}_k$ and $q = \mathcal{O}(h^{k+1})$. Then

$$(t_h u)_i = (t_h(p+q))_i = (t_h p)_i + (t_h q)_i = (t_h q)_i.$$

Each element of the vector $t_h q$ has the form

$$\sum_{j \in \mathcal{D}_i} \alpha_j q_j - \sum_{j \in \mathcal{E}_i} \beta_j (Lq)_j$$

and since $|Lq| \leq \|L\||q| = \|L\|\mathcal{O}(h^{k+1}) = \mathcal{O}(h^{k+1})$ because L has bounded coefficients where $\|L\|$ is a constant independent of h. Hence

$$\begin{aligned} \left| \sum_{j \in \mathcal{D}_i} \alpha_j q_j - \sum_{j \in \mathcal{E}_i} \beta_j (Lq)_j \right| &\leq \sum_{j \in \mathcal{D}_i} |\alpha_j||q_j| + \sum_{j \in \mathcal{E}_i} |\beta_j||(Lq)_j| \\ &\leq \sum_{j \in \mathcal{D}_i} |\mathcal{O}(h^{-2})||\mathcal{O}(h^{k+1})| + \sum_{j \in \mathcal{E}_i} \mathcal{O}(1)|\mathcal{O}(h^{k+1})| \\ &= |\mathcal{O}(h^{k-1})| \end{aligned}$$

Thus $\|t_h u\| = \mathcal{O}(h^{k-1})$. •

The next theorem is a direct result of Theorem 3.15.

Theorem 3.16 *A HODIEX discretization exact on $\mathcal{P}_k$ is consistent if $k \geq 3$.*

Assume a non-trivial *HODIEX* discretization exists exact on $\mathcal{P}$. From (3.38) we see that the truncation and discretization errors are related by

$$e_h g = A^{-1} t_h u.$$

So if

$$\|t_h u\| = \mathcal{O}(h^k)$$

then

$$\begin{aligned} \|e_h g\| &= \|A^{-1} t_h u\| \\ &\leq \|A^{-1}\| \|t_h u\| \\ &\leq B\mathcal{O}(h^k) \quad \text{by the assumed stability} \\ &= \mathcal{O}(h^k). \end{aligned}$$

Hence we have the following.

Theorem 3.17 *If an exact non-trivial stable HODIEX discretization exists on a given finite dimensional linear space, then the truncation error and discretization error are of the same order.*

Thus while consistency is easy to prove, the actual convergence of the *HODIEX* discretizations exact on polynomial spaces depends on stability, which is not easy to prove for general operators.

3.4 Summary

It has been shown that the discretization given by (3.5) has various desirable properties ([79], [122], [5]). We have shown that *HODIEX* discretizations, of any order, are $\mathcal{O}(h)$ perturbations of the $\mathcal{O}(h^2)$ finite difference stencil given by (3.5) and for sufficiently small h, the coefficient matrix of a linear system generated by *HODIEX* has similar properties as one generated by (3.5) [90].

For simple operators we have shown that there are theoretical upper limits to the order of accuracy that may be obtained with a *HODIEX* discretization. If k is large enough so that $\mathcal{P}_k \cap \mathcal{N}_L$ is linearly independent with respect to the nine stencil points, then the *HODIEX* discretization fails to exist. If a stable *HODIEX* discretization does exist exact on a linear space, then the truncation error has the same order as the discretization error.

Chapter 4

THE PRECONDITIONER

For sufficiently small h the coefficient matrix A arising from a *HODIEX* discretization has the general properties required for convergence of most iterative methods (diagonal dominance, positive definiteness, etc.); however, the *HODIEX* discretization does not guarantee a favorable spectral radius. Since the rate of convergence depends on the spectral properties of the coefficient matrix, convergence may be slow. We use domain decomposition to construct an easily parallelized preconditioner with which the linear system $Au = b$ is transformed into a system which has the same solution, but more favorable spectral properties, thus yielding rapid convergence. In this chapter the construction of our preconditioner using domain decomposition is discussed in detail.

4.1 DD Preconditioning

The preconditioner P is constructed as a domain decomposition (DD) approximation of A. The DD approach not only results in an easily factored preconditioner whose inverse action is easy to compute, but also both of these operations are highly parallelizable. Both left and right preconditioning are possible, but in this dissertation we only consider right preconditioning. For a right preconditioner P we first solve the system

$$(AP^{-1})v = b, \tag{4.1}$$

and then solve

$$Pu_h = v \tag{4.2}$$

to get the approximation u_h of u.

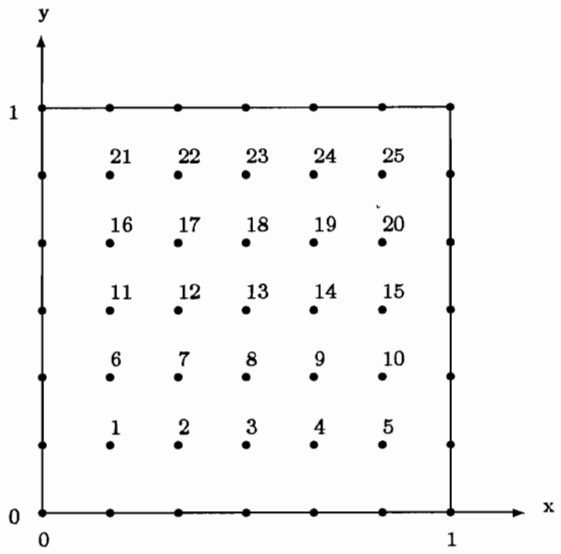


Figure 4.1: A 5×5 grid.

4.1.1 The Global Domain

We assume that the domain is the unit square on which a rectangular mesh has been imposed. As shown in Figure 4.1, the nodes (grid points) are numbered using a natural ordering of the equations (left to right and bottom to top). The *HODIEX* discretization of the PDE on the global domain results in a coefficient matrix of the form shown in Figure 4.2. Observe that A is banded, block tridiagonal, and each block tridiagonal.

Once the desired domain decomposition has been selected, the first step in the construction of the preconditioner P is to reorder the grid points by subdomain. A *HODIEX* discretization is done on each subdomain to construct P and A is reordered to conform to the ordering of grid points used for P. The method by which domain decomposition is used to construct P will be demonstrated using several examples. First, we consider a 5×5 grid (25 interior grid points) placed over the domain as shown in Figure 4.1.

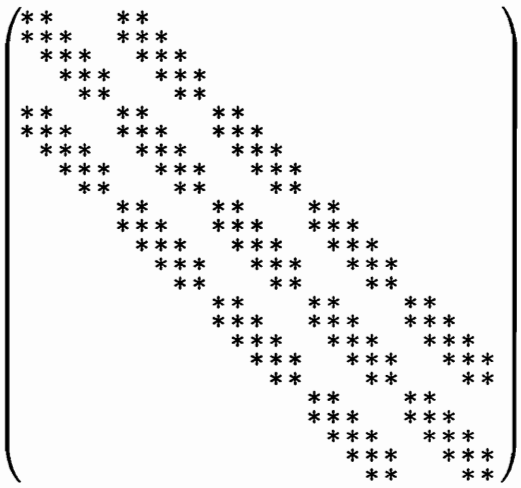


Figure 4.2: The coefficient matrix for a 5×5 grid.

4.1.2 The Domain Decomposition

We will give a detailed description of a 2×2 domain decomposition shown in Figure 4.3 followed by several less detailed examples.

2 x 2 Decomposition

Domain decomposition involves dividing the domain into rectangles by drawing equally spaced horizontal and/or vertical interfaces (lines) across the domain by connecting horizontal or vertical sets of nodes (grid points). The distance between adjacent horizontal interface lines is constant H_y and the distance between adjacent vertical interface lines is constant H_x. Figure 4.3 shows a 5×5 grid divided into a 2×2 decomposition in which $h_x = h_y = h = \frac{1}{6}$ and $H_x = H_y = H = \frac{1}{2}$.

The one horizontal interface line and one vertical interface line divide the domain into four distinct types of subdomains: a crosspoint subdomain, two y-interface subdomains, two x-interface subdomains, and four interior subdomains. Many authors have used the term *subdomain* to refer only to what we refer to as an *interior subdomain*. Our definition of

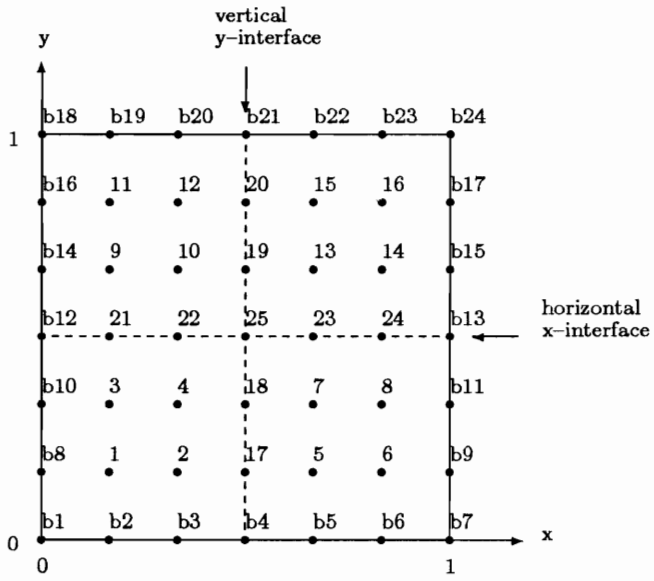


Figure 4.3: A 5×5 grid with a 2×2 domain decomposition.

subdomain is somewhat more general in that it includes points on interface lines as members of special subdomains in their own right.

Crosspoint subdomain. Grid points on the intersection of horizontal and vertical interface lines are called crosspoints and collectively constitute one *crosspoint subdomain*. The crosspoint subdomain has a coarse mesh size in both x and y directions of $H_x = \frac{1}{2}$ and $H_y = \frac{1}{2}$. In Figure 4.3 the crosspoint subdomain consists of only the one point $\{25\}$ (the 3×3 decomposition shown in Figure 4.6 shows a crosspoint subdomain with four grid points). The boundary of the crosspoint subdomain consists of grid points on the intersection of the interface lines with the natural boundary and the four corners. The boundary for crosspoint subdomain $\{25\}$ consists of the eight natural boundary points $\{b1, b4, b7, b12, b13, b18, b21, b24\}$.

Y-interface subdomains. Grid points which lie only on vertical interface lines are called y-interface points. The y-interface points between adjacent parallel horizontal inter-

faces or between the natural boundary and an adjacent parallel horizontal interfaces constitute one *y-interface subdomain.* In Figure 4.3 the two y-interface subdomains each contain just one interface line segment and contain only two grid points, {17, 18} and {19, 20} (the 3×3 decomposition shown in Figure 4.6 shows interface subdomains with multiple interface line segments). The points in a y-interface subdomain have a fine mesh size in the vertical direction and a coarse mesh size in the horizontal direction; thus for this example the node spacing is $H_x = \frac{1}{2}$ and $h_y = \frac{1}{6}$. The boundary of y-interface subdomains consists of grid points on the intersection of the vertical interface lines with horizontal interface lines (crosspoints) and points on the natural boundary in the horizontal direction. The boundary for y-interface subdomain {17, 18} consists of crosspoint {25} and nine natural boundary points {$b1$, $b4$, $b7$, $b8$, $b9$, $b10$, $b11$, $b12$, $b13$}. Notice that there is no contact with x-interface subdomains (described next).

X-interface subdomains. Grid points which lie only on horizontal interface lines are called x-interface points. The x-interface points between adjacent parallel vertical interfaces or between the natural boundary and an adjacent parallel vertical interfaces constitute one *x-interface subdomain.* In Figure 4.3 the two x-interface subdomains each contain only two grid points, {21, 22} and {23, 24}. The points in an x-interface subdomain have a fine mesh size in the horizontal direction and a coarse mesh size in the vertical direction; thus for this example the node spacing is $H_x = \frac{1}{6}$ and $h_y = \frac{1}{2}$. The boundary of x-interface subdomains consists of crosspoints and points on the natural boundary in the vertical direction. The boundary for x-interface subdomain {21, 22} consists of crosspoint {25} and nine natural boundary points {$b1$, $b2$, $b3$, $b4$, $b12$, $b18$, $b19$, $b20$, $b21$}. Notice that there is no contact with y-interface subdomains.

Interior Subdomains. Grid points not on any interface line are called interior points. The interface lines divide the global domain into a number of disjoint sets of such points called *interior subdomains.* In Figure 4.3 there are four interior subdomains

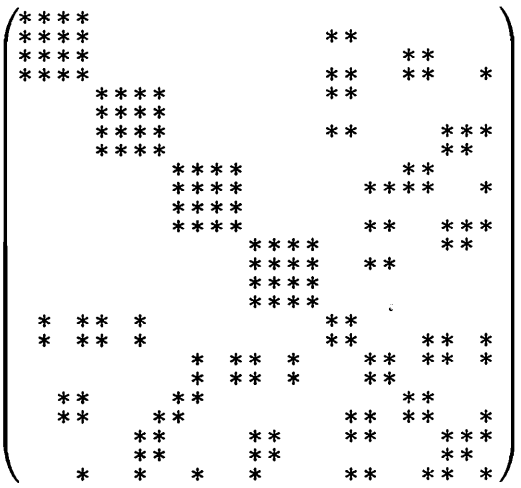


Figure 4.4: A 5×5 grid coefficient matrix after 2×2 DD reordering.

each containing four grid points. The node spacing between points in an interior subdomain is always the fine mesh size h_x and h_y. In this example the four interior subdomains are the point sets {1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}, and {13, 14, 15, 16}. The boundary of interior subdomains consists of grid points on interface lines and natural boundary points if the subdomain is adjacent to the natural boundary. In this 2×2 DD example each interior has natural boundary points; but in larger decompositions, such as the 3×3 decomposition and 4×3 decomposition which follow, there are interior domains with no natural boundary points. The boundary for interior subdomain {1, 2, 3, 4} consists of interface points {17, 18, 21, 22}, crosspoint {25}, and seven natural boundary points {$b1$, $b2$, $b3$, $b4$, $b8$, $b10$, $b12$}.

Before grid points are renumbered, the subdomains themselves must be numbered. This numbering is determined by using natural ordering on the interiors first, then the y-interfaces, then the x-interfaces, and last the crosspoint subdomain. Once the individual subdomains are numbered the individual grid points within subdomains are numbered using natural ordering by subdomain number.

When the DD reordering shown in Figure 4.3 is applied to the global coefficient matrix

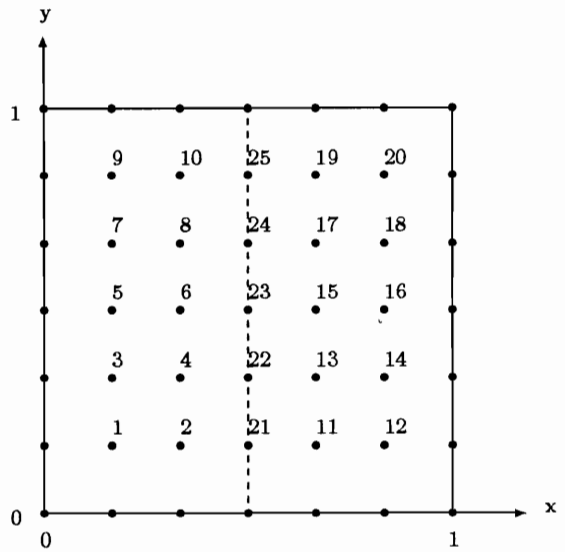


Figure 4.5: A 5×5 grid with a 2×1 strip domain decomposition.

A, it results in the coefficient matrix shown in Figure 4.4.

2 x 1 Strip Decomposition

Figure 4.5 is an example of a 2×1 decomposition which yields two interior strips.

Interior Subdomains. The two interior subdomains consist of the point sets {1, 2, 3, 4, 5, 6, 7, 8, 9, 10} and {11, 12, 13, 14, 15, 16, 17, 18, 19, 20}. Interior subdomains have a fine mesh size of $h_x = \frac{1}{6}$ and $h_y = \frac{1}{6}$.

Y-interface subdomains. There is only one y-interface subdomain consisting of the point set {21, 22, 23, 24, 25}. The y-interface subdomain has the a coarse mesh in the x direction $H_x = \frac{1}{2}$ and a fine mesh in the y direction $h_y = \frac{1}{6}$.

X-interface subdomains. There are no x-interface subdomains.

Crosspoint Subdomain. There is no crosspoint subdomain.

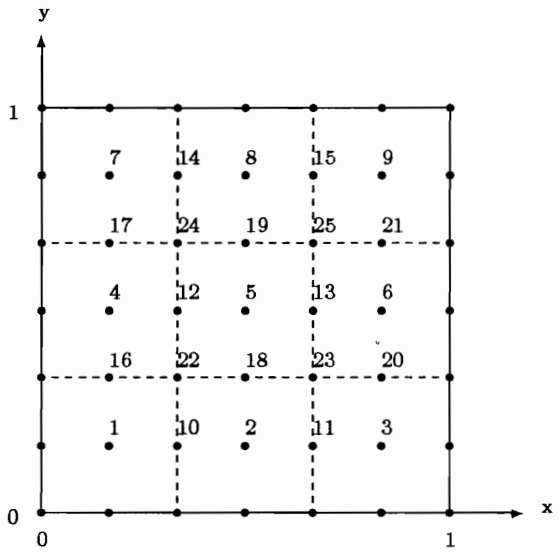


Figure 4.6: A 5×5 grid with a 3×3 domain decomposition.

3 x 3 Decomposition

Figure 4.6 is an example of a 3×3 decomposition.

Interior Subdomains. There are nine interior subdomains consisting of the point sets {1}, {2}, {3}, {4}, {5}, {6}, {7}, {8}, and {9}. The interior subdomain consisting of grid point {5} has no natural boundary points. All interior subdomains have $h = h_x = h_y = \frac{1}{6}$.

Y-interface subdomains. There are three y-interface subdomains consisting of the points sets {10, 11}, {12, 13}, and {14, 15}. All y-interface subdomains have $h_x = \frac{1}{3}$ and $h_y = \frac{1}{6}$.

X-interface subdomains. There are three x-interface subdomains consisting of the points sets {16, 17}, {18, 19}, and {20, 21}. All x-interface subdomains have $h_x = \frac{1}{6}$ and $h_y = \frac{1}{3}$.

Crosspoint Subdomain. The crosspoint subdomain consists of the 4 point set {22, 23, 24, 25}. The crosspoint subdomain has $h = h_x = h_y = \frac{1}{3}$.

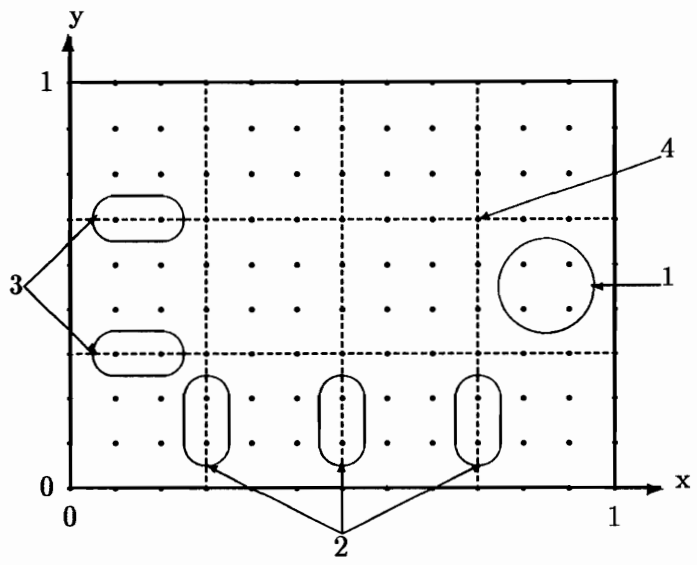


Figure 4.7: An 11×8 grid with 12 interior subdomains.

4 x 3 decomposition on a 11 x 8 grid

Figure 4.7 shows a slightly more complex 11×8 grid in which $h_x \neq h_y$.

Interior Subdomains. There are twelve interior subdomains. Two of the interior subdomains have no natural boundary points (all boundary points are interface points). An examples interior subdomain is indicated by number 1.

***Y*-interface subdomains.** There are three y-interface subdomains consisting of grid points from multiple interface line segments. An example y-interface subdomain is indicated by number 2.

***X*-interface subdomains.** There are four x-interface subdomains consisting of grid points from multiple interface line segments. An example x-interface subdomain is indicated by number 3.

Crosspoint Subdomain. The crosspoint subdomain consists of 6 grid points. The 4 indicates one of the six points in the crosspoint subdomain.

4.1.3 The DD Preconditioner

Once the subdomain structure has been determined and the grid points have been renumbered, A is reordered corresponding to the grid point reordering. No other changes are made to A. After this reordering A has a similar structure as before except for a smaller bandwidth in each subdomain (the matrix example shown in Figure 4.4 does not show this structure because the number of grid points within subdomains is too small).

The preconditioner matrix P is computed by discretizing each subdomain using *HODIEX* on the same space $\mathcal{P}_k$ as used for A and the 3×3 compact stencil with mesh size determined by the subdomain. Similarly the boundary is determined by subdomain.

Interior subdomains. The interiors use the same fine grain discretization (h_x in the x direction and h_y in the y direction) as was used for A and thus no additional computations are involved. Boundaries are points on the interfaces or the natural boundary if it is adjacent.

Y-interface subdomains. A coarse/fine discretization is used (H_x in the x direction and h_y in the y direction). Right and left boundaries are x-interfaces or the natural boundary if it is adjacent. Upper and lower boundaries are the natural boundaries.

X-interface subdomains. A fine/coarse discretization is used (h_x in the x direction and H_y in the y direction). Upper and lower boundaries are y-interfaces or the natural boundary if it is adjacent. Right and left boundaries are the natural boundaries.

Crosspoint subdomain. The crosspoint discretization is coarse in both directions (H_x in the x direction and H_y in the y direction). The boundary is the natural boundary.

The result of these discretizations is the linear system

$$Pu_H = c$$

whose solution actually corresponds to an approximate solution to the PDE, although to a lesser accuracy than the global fine grain discretization. The order of accuracy of the

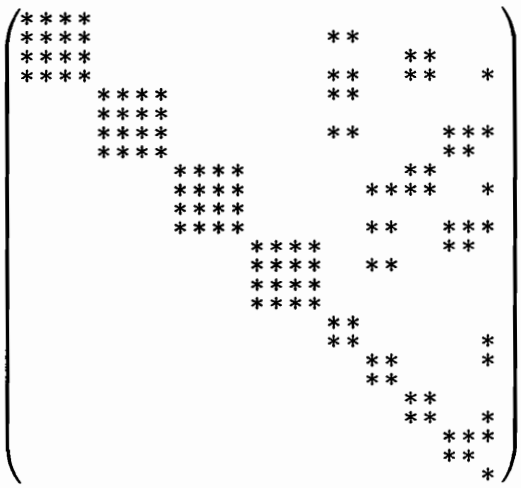


Figure 4.8: The preconditioner with 5×5 grid and 2×2 DD reordering.

preconditioner as a solution to the PDE increases as the number of subdomains is increased. We take the matrix P as our preconditioner and we take u_H as our initial guess. The structure of P for the simple 5×5 grid with a 2×2 domain decomposition is shown in Figure 4.8. Notice that P is block upper triangular.

The general form of the resulting coefficient matrix P within interiors of subdomains is again block tridiagonal with a bandwidth determined by the width of the subdomain. The example preconditioner matrix P shown in Figure 4.8 does not (as in the case for A) display the block tridiagonal structure due to the small number of grid points within subdomains. As the number of interior subdomains increases, the bandwidth of the interior subdomain systems decreases, with a corresponding decrease in the time to solve for interior subdomains; however, there is also a corresponding increase in the time to solve the increased number of interface subdomains and the larger crosspoint subdomain system.

4.2 The DD Algorithm

Our domain decomposed preconditioned iterative algorithm may be summarized as follows:

1. A rectangular mesh is placed over the global domain (thus fixing the constants h_x and h_y). A particular domain decomposition is chosen by drawing horizontal and vertical interface lines over the domain (thus fixing the constants H_x and H_y).

2. The chosen DD determines which grid points lie in which subdomains. After numbering the subdomains, the grid points are renumbered by subdomain accordingly.

3. A global fine grid discretization of the PDE using *HODIEX* results in the linear system $Au_h = b$. A and b are then reordered to correspond to the chosen decomposition. A is sparse and a matrix-vector multiply parallelizes well.

4. A fine/coarse DD discretization of the PDE using *HODIEX* results in another linear system $Pu_H = c$ which also solves the PDE, although to a lesser degree of accuracy than $Au_h = b$. No additional computation is required on interior points since for interior points A and P are identical. P is easily factored and its inverse action may rapidly be computed using the factors. An initial guess $u_H = P^{-1}c$ is computed whose accuracy is as high as $\mathcal{O}(h^{k/2})$ if $H^2 = h$ and the space $\mathcal{P}_k$ is used (see Chapter 5).

5. An iterative solver is applied which takes full advantage of the sparsity of A. There is no resultant fill-in in A. Only the matrix-vector multiply Ax and application of the inverse of P are required. Using the factored preconditioner with initial guess u_H, we achieve both rapid convergence and high accuracy.

4.3 Matrix Structure

The DD approach described above may be easily viewed through matrix notation. The nonsymmetric reordered global coefficient matrix has the form

$$A = \begin{pmatrix} A_I & A_{IB} & A_{IC} \\ A_{BI} & A_B & A_{BC} \\ A_{CI} & A_{CB} & A_C \end{pmatrix}, \tag{4.3}$$

where the large submatrix A_I corresponds to the interiors, A_{IB} and A_{BI} represent interactions between subdomains and interfaces, A_B corresponds to the interfaces, A_{BC} and

A_{CB} contain interactions between interfaces and cross points, A_C corresponds to the cross points, and A_{IC} and A_{CI} involve coupling of interiors with cross points. The size of the sub-matrices may vary greatly in size. In the case of strips the sub-matrices with crosspoint interactions are zero and A has the simpler form

$$A = \begin{pmatrix} A_I & A_{IB} \\ A_{BI} & A_B \end{pmatrix}. \tag{4.4}$$

The structure of the preconditioner is

$$P = \begin{pmatrix} A_I & A_{IB} & A_{IC} \\ 0 & \tilde{A}_B & \tilde{A}_{BC} \\ 0 & 0 & \tilde{A}_C \end{pmatrix}. \tag{4.5}$$

P is a block triangular approximation of A and both A and P contain the large block tridiagonal submatrix A_I. If $h_x = h_y = h$ and $H_x = H_y = H$ it may be observed that AP^{-1} has the form [66]

$$AP^{-1} = \begin{pmatrix} I_h & - \\ - & - \end{pmatrix} \tag{4.6}$$

where AP^{-1} is a $(\frac{1}{h}-1)^2 \times (\frac{1}{h}-1)^2$ matrix and I_h is a $(\frac{1}{h}-\frac{1}{H})^2 \times (\frac{1}{h}-\frac{1}{H})^2$ identity matrix, a fact which may be exploited to reduce the cost of an iteration of the solver (the current version of our code does not take advantage of this optimization).

The block upper triangular structure of P makes its inverse action easy to compute. To compute $P^{-1}v$ we solve $Pw = v$ or

$$\begin{pmatrix} A_I & A_{IB} & A_{IC} \\ 0 & \tilde{A}_B & \tilde{A}_{BC} \\ 0 & 0 & \tilde{A}_C \end{pmatrix} \begin{pmatrix} w_I \\ w_B \\ w_C \end{pmatrix} = \begin{pmatrix} v_I \\ v_B \\ v_C \end{pmatrix}. \tag{4.7}$$

The first step is the cross point solve

$$\tilde{A}_C w_C = v_C, \tag{4.8}$$

followed by the interface solves

$$\tilde{A}_B w_B = v_B - \tilde{A}_{BC} w_C, \tag{4.9}$$

and finally the independent interior solves

$$A_I w_I = v_I - A_{IC} w_C - A_{IB} w_B. \tag{4.10}$$

4.3.1 Summary

The proposed preconditioner P is a domain decomposition approximation of A. P is block upper triangular, easily factored, and its application is a highly parallelizable operation.

Since A_I is large and block tridiagonal with disjoint blocks, the interior solves are a highly parallelizable operation in which each solve may be done in parallel. The independent interior solves consist of h-scale discretizations on the subdomains of the PDE. Similarly, the interface solves are a highly parallelizable operation since the interface subdomains are also disjoint. The independent interface subdomain solves consist of $h \times H$-scale discretizations on both the x- and y-interface subdomains.

The only parallelization bottleneck is the cross point solve. The cross point operator $\tilde{A}_C$ is just an H-scale coarse grid discretization of the original PDE on which we use a banded Gaussian elimination direct solver. This requires global communication on distributed memory machines and is a possible sequential bottleneck on any parallel machine; however, the crosspoint solve is generally very small and thus a minimal bottleneck results. We use exact solves on all subdomains which eliminates the question as to how much only approximate subdomain solves would penalize the overall iterative convergence.

Chapter 5

PRECONDITIONER THEORY

In this chapter we prove various properties of AP^{-1} where A is the discrete linear operator resulting from a global fine h mesh discretization and P, the preconditioner, is the discrete linear operator resulting from a *HODIEX* domain decomposition discretization which includes both fine h and coarse H mesh sizes. Classical *GMRES* theory gives general convergence rate estimates in terms of residual norms of the unpreconditioned system. We give convergence rate estimates in terms of the preconditioned system, and then by considering the properties of the preconditioner, we show sharper convergence rate estimates.

5.1 Properties of the Preconditioner

The discrete operator A is assumed to be reordered to correspond to the grid point numbering used for P. In the following we use the notation $[\mathcal{O}(h^k)]$ to denote an N–vector for which each element is $\mathcal{O}(h^k)$.

Assume a fine grain $\mathcal{O}(h^4)$ accurate *HODIEX* discretization of the PDE

$$au_{xx} + cu_{yy} + du_x + eu_u + fu = g$$

resulting in the linear system

$$Au_h = g_h \tag{5.1}$$

in which

$$u_h = r_h u + [\mathcal{O}(h^4)] \tag{5.2}$$

and

$$g_h = r_h g + [\mathcal{O}(h)]. \tag{5.3}$$

Also assume a $\mathcal{O}(H^4)$ *HODIEX* DD discretization resulting in the linear system

$$Pu_H = g_H \tag{5.4}$$

in which

$$u_H = r_h u + [\mathcal{O}(H^4)] \tag{5.5}$$

and

$$g_H = r_h g + [\mathcal{O}(H)]. \tag{5.6}$$

We may write

$$A = P + N \tag{5.7}$$

where N is a perturbation operator. As mentioned previously, all norms are the supremum norm. Notice that the norm

$$\|A\| \equiv \max_{\substack{\|r_h u\|=1 \\ u \in C^2}} \|A r_h u\|_\infty$$

is really just

$$\|A\| = \max_{\substack{\|x\|=1 \\ x \in \mathcal{R}^N}} \|Ax\|_\infty.$$

The following theorem relates the linear discrete operator represented by A to its domain decomposition approximation P.

Theorem 5.1

$$\|N\| = \|A - P\| = \mathcal{O}(H). \tag{5.8}$$

Proof: Assume $u \in \mathcal{C}^2(\mathcal{D})$ and $\|r_h u\| = 1$. Subtracting (5.4) from (5.1) yields

$$\begin{aligned} Au_h - Pu_H &= g_h - g_H \\ &= [\mathcal{O}(h)] - [\mathcal{O}(H)] \quad \text{from (5.3) and (5.6)} \\ &= [\mathcal{O}(H)] \quad \text{since } h < H. \end{aligned}$$

Substituting from (5.2) and (5.5) then gives

$$A(r_h u + [\mathcal{O}(h^4)]) - P(r_h u + [\mathcal{O}(H^4)]) = [\mathcal{O}(H)] \tag{5.9}$$

or

$$(A - P) r_h u = P[\mathcal{O}(H^4)] - A[\mathcal{O}(h^4)] + [\mathcal{O}(H)]. \tag{5.10}$$

In Theorem 3.6 we saw that any *HODIEX* discretization exact on $\mathcal{P}_k$, $k \geq 4$, results in an $\mathcal{O}(h)$ relative perturbation (or $\mathcal{O}(H)$ for an H scale discretization) of Krylov's [79] 9-point compact central finite difference stencil

$$\frac{1}{12h^2}\begin{array}{|c|c|c|}\hline a+c & 10c-2a-6he & a+c \\ \hline 10a-2c+6hd & -20(a+c)+12h^2 f & 10a-2c-6hd \\ \hline a+c & 10c-2a+6he & a+c \\ \hline\end{array} \quad u = g + \mathcal{O}(h^2). \tag{5.11}$$

Thus for any element of $P[\mathcal{O}(H^4)]$ we have

$$\begin{aligned}\Big(P[\mathcal{O}(H^4)]\Big)_i &= \left(\sum_{j=1}^{9} \alpha_j \left(1 + \mathcal{O}(H)\right) \mathcal{O}(H^4)\right) \\ &= \left(\sum_{j=1}^{9} \alpha_j\right) \left(1 + \mathcal{O}(H)\right) \mathcal{O}(H^4) \\ &= \mathcal{O}(H) \left(1 + \mathcal{O}(H)\right) \mathcal{O}(H^4) \\ &= \mathcal{O}(H),\end{aligned}$$

where the $\alpha_j(1 + \mathcal{O}(H))$ are relative perturbations of the entries in (5.11) evaluated at the appropriate grid points and the PDE coefficients are expanded by Taylor series about the central stencil point yielding $a_2 = a_1 + \mathcal{O}(H)$, $a_3 = a_1 + \mathcal{O}(H)$, etc. We have assumed that the coefficients are C^2 functions with uniformly bounded derivatives in $\mathcal{D}$. With these assumptions it is easy to show that $\sum_{j=1}^{9} \alpha_j = \mathcal{O}(H)$. In a similar manner for each element of $A[\mathcal{O}(h^4)]$ we have

$$\Big(A[\mathcal{O}(h^4)]\Big)_i = \mathcal{O}(h). \tag{5.12}$$

Applying these results to (5.10) yields

$$(A - P) r_h u = [\mathcal{O}(H)] - [\mathcal{O}(h)] + [\mathcal{O}(H)] = [\mathcal{O}(H)] \tag{5.13}$$

and thus

$$\|(A - P)r_h u\| = \mathcal{O}(H) \tag{5.14}$$

which yields

$$\|A - P\| = \mathcal{O}(H). \tag{5.15}$$

•

Definition 5.1 *The matrix A is said to be positive definite with respect to a linear space $\mathcal{S}$ if $v^T Av > 0$ for every nonzero $v \in \mathcal{S}$.*

With the previous theorem it is easy to show that AP^{-1} is positive definite with respect to solution vectors.

Theorem 5.2 *Assume a nonzero $g \in \mathcal{C}^2(\mathcal{D})$. Let $v = A^{-1} I_h s_h g$. Then for sufficiently small H (or h since $H = mh$), $v^T Av > 0$.*

Proof: Assume $v = A^{-1} I_h s_h g > 0$. Then

$$\begin{aligned} v^T(AP^{-1})v &= v^T(I + NP^{-1})v \\ &= v^T v + v^T N P^{-1} v. \end{aligned}$$

Clearly $v^T v$ is positive. By the assumed stability of *HODIEX* and the fact that both A and P are *HODIEX* discretizations and $g \in \mathcal{C}^2(\mathcal{D})$ we know that $\|v\|$ and $\|P^{-1}\|$ are bounded independent of H. Thus the second term can be made positive by reducing H and making $\|N\|$ as small as desired by Theorem 5.1. •

We would like AP^{-1} to be close to the identity operator. If we fix h and let H get smaller the following theorem shows that this is indeed the case.

Theorem 5.3

$$\|AP^{-1} - I\| = \mathcal{O}(H) \tag{5.16}$$

and

$$\|PA^{-1} - I\| = \mathcal{O}(H). \tag{5.17}$$

Proof:

$$\begin{aligned} \|(AP^{-1} - I)v\| &\leq \|(A - P)\| \, \|P^{-1}\| \, \|v\| \\ &= \mathcal{O}(H)C \, \|v\| \\ &= \mathcal{O}(H) \end{aligned}$$

by Theorem 5.1 and the assumed stability of P. Interchanging A and P proves the second equality. •

The following theorem allows us to estimate the condition number of AP^{-1}.

Theorem 5.4

$$\|AP^{-1}\| \leq 1 + \mathcal{O}(H) \tag{5.18}$$

and

$$\|(AP^{-1})^{-1}\| \leq 1 + \mathcal{O}(H). \tag{5.19}$$

Proof: Assume a nonzero v.

$$\begin{aligned} \|AP^{-1}v\| - \|v\| &\leq \|AP^{-1}v - v\| \\ &= \|(AP^{-1} - I)v\| \\ &\leq \|AP^{-1} - I\| \, \|v\|. \end{aligned}$$

Thus

$$\begin{aligned}\frac{\|AP^{-1}v\|}{\|v\|} &\leq 1 + \|AP^{-1} - I\| \\ &= 1 + \mathcal{O}(H)\end{aligned}$$

from which it follows that

$$\|AP^{-1}\| \leq 1 + \mathcal{O}(H).$$

Interchanging A and P in this proof gives the second inequality. •

Applying Theorem 5.4 we get the following estimate of the condition number of AP^{-1}.

Theorem 5.5

$$\mathcal{K}(AP^{-1}) \leq 1 + \mathcal{O}(H). \tag{5.20}$$

It is clear from the construction of the operators A and P that they differ on non-interior grid points; however, our real interest is in how closely P^{-1} approximates A^{-1}.

Theorem 5.6

$$\|A^{-1} - P^{-1}\| = \mathcal{O}(H). \tag{5.21}$$

Proof: By the assumed stability of the *HODIEX* discretization, both $\|A^{-1}\|$ and $\|P^{-1}\|$ are bounded independent of h or H. Let $Q = I - AP^{-1}$, then

$$\|I - Q\| = \|AP^{-1}\| = 1 + \mathcal{O}(H)$$

by Theorem 5.4. By Theorem 5.3 $\|Q\| = \mathcal{O}(H) < 1$ so by Householder [75]

$$\|(I - Q)^{-1}\| \leq \frac{1}{1 - \mathcal{O}(H)}.$$

Since

$$\begin{aligned}P^{-1} &= A^{-1}(AP^{-1}) \\ &= A^{-1}(I - Q)\end{aligned}$$

then

$$\begin{aligned} A^{-1} &= P^{-1}(I-Q)^{-1} \\ &= P^{-1}(I+Q+Q^2+...) \\ &= P^{-1}+P^{-1}Q(I-Q)^{-1} \end{aligned}$$

from which it follows that

$$A^{-1}-P^{-1} = P^{-1}Q(I-Q)^{-1}.$$

Hence

$$\begin{aligned} \|A^{-1}-P^{-1}\| &\leq \|P^{-1}\|\,\|Q\|\,\|(I-Q)^{-1}\| \\ &= \|P^{-1}\|\frac{\mathcal{O}(H)}{1-\mathcal{O}(H)} \\ &= \mathcal{O}(H) \end{aligned}$$

for sufficiently small H. •

Before considering how the above theory affects *GMRES*, we summarize the main results derived so far. For sufficiently small H:

- $\|A-P\| = \mathcal{O}(H)$.
- The preconditioned system AP^{-1} retains the positive definiteness of A.
- $\mathcal{K}(AP^{-1}) \leq 1+\mathcal{O}(H)$
- $\|A^{-1}-P^{-1}\| = \mathcal{O}(H)$.

5.2 GMRES

We use *GMRES* of Saad and Schultz [103] as the iterative solver of the linear system. *GMRES* is a generalized minimum residual Krylov subspace method. The theory on *GMRES* can be found in Eisenstat, Elman, and Schultz [57].

Definition 5.2 *For a linear operator $A : \mathcal{R}^N \rightarrow \mathcal{R}^N$ and a vector $v \in \mathcal{R}^N$ a Krylov subspace is defined by*

$$\mathcal{K}_i(A, v) \equiv span\{v, Av, \ldots, A^{i-1}v\}. \tag{5.22}$$

Let A be a nonsymmetric linear operator on $\mathcal{R}^N$. Given $b \in \mathcal{R}^N$, we use *GMRES* to solve the system

$$Au_h = b. \tag{5.23}$$

An initial guess u_0 (usually zero for most solvers) is used to compute the initial residual

$$r_0 = b - Au_0. \tag{5.24}$$

If $u_0 \neq 0$, Equation 5.23 can be transformed into an equivalent problem which does have initial guess 0 by

$$\begin{aligned} Av &= A(u - u_0) \\ &= Au - Au_0 \\ &= b - Au_0 \\ &= r_0. \end{aligned}$$

GMRES uses the subspace

$$\mathcal{V}_i = \mathcal{K}_i(A, r_0) \tag{5.25}$$

to solve the problem

$$Av = r_0 \tag{5.26}$$

where r_0 is the initial residual defined by (5.24). In the ith iteration a vector v_i is computed from the Krylov subspace

$$\mathcal{K}_i(A, r_0) = span\{r_0, Ar_0, \ldots, A^{i-1}r_0\} \tag{5.27}$$

which minimizes the norm of the residual

$$\|r_i\| = \|b - Av_i\|. \tag{5.28}$$

It can be shown that in exact arithmetic no more than N iterations are required to drive the residual to zero [57]. The residual in Equation (5.28) can be in any consistent norm [36].

5.2.1 Properties of the Residuals

Denote by Π_k the space of polynomials of maximum degree k. The following theorem by Heinkenschloss [71] is a direct consequence of the definition of a Krylov subspace and will be used in proofs of succeeding theorems.

Theorem 5.7 *The Krylov subspace defined by Equation (5.22) may be characterized by*

$$\mathcal{K}_i(A, v) = \{\pi(A)v : \pi \in \Pi_{i-1}\}.$$

Assume a fine grain *HODIEX* discretization results in the system

$$Au_h = b \tag{5.29}$$

and a DD *HODIEX* discretization results in the system

$$Pu_H = c. \tag{5.30}$$

We use the initial guess $u_H = P^{-1}c$ to translate the problem into the form given in Equation (5.26) so that after solution for v we set $u = v + u_H$. We use *GMRES* to solve the preconditioned system

$$AP^{-1}w = r_0 \tag{5.31}$$

where

$$Pv = w. \tag{5.32}$$

Let p_k, $q_k \in \Pi_k$ and $q_k(0) = 1$. By 5.7 *GMRES* generates iterates

$$v_k \in \mathcal{K}_k(AP^{-1}, r_0) = \{p(AP^{-1})r_0 \quad | \quad p \in \Pi_{k-1}\} \tag{5.33}$$

with the minimum residual. Hence for the kth iteration, v_k must be found to minimize the residual, where

$$v_k = p_{k-1}(AP^{-1})r_0 \tag{5.34}$$

for some $p_{k-1} \in \Pi_{k-1}$.

The residual r_k may be written as

$$\begin{aligned} r_k &= r_0 - AP^{-1}v_k \\ &= r_0 - AP^{-1}p_{k-1}(AP^{-1})r_0 \\ &= (I - p_k(AP^{-1}))r_0 \\ &= q_k(AP^{-1})r_0, \end{aligned}$$

for some p_k, $q_k \in \Pi_k$. Thus the residual in *GMRES* must satisfies

$$\|r_k\| = \min_{q_k} \|q_k(AP^{-1})r_0\| \tag{5.35}$$

$$\leq \min_{q_k} \|q_k(AP^{-1})\| \, \|r_0\|. \tag{5.36}$$

Saad and Schultz show that *GMRES* produces the smallest possible residual for a given number of iterations.

By factoring the polynomial $q_k(AP^{-1}) = q_1(AP^{-1})q_{k-1}(AP^{-1})$ we get the following theorem.

Theorem 5.8

$$\|r_k\| \leq \min_{q_1} \|q_1(AP^{-1})\| \, \|r_{k-1}\|. \tag{5.37}$$

Proof:

$$\begin{aligned} \|r_k\| &= \min_{q_k} \|q_k(AP^{-1})r_0\| \\ &\leq \min_{q_1, q_{k-1}} \|q_1(AP^{-1})q_{k-1}(AP^{-1})r_0\| \\ &\leq \min_{q_1} \|q_1(AP^{-1})\| \quad \min_{q_{k-1}} \|q_{k-1}(AP^{-1})r_0\| \\ &= \min_{q_1} \|q_1(AP^{-1})\| \, \|r_{k-1}\|. \end{aligned}$$

•

5.2.2 Classical GMRES Theory

Eisenstat, Elman, and Schultz [57] characterize the rate of convergence of *GMRES* in terms of the minimal eigenvalue of the symmetric part of the operator A and the norm of the operator. These two quantities may be defined by

$$c_A = \inf_{v \neq 0} \frac{(v, Av)}{(v, v)}$$

and

$$C_A = \sup_{v \neq 0} \frac{\|Av\|}{\|v\|}$$

respectively. They characterize the decrease in the norm of the residual in a single step as follows.

Theorem 5.9 *If $c_A > 0$, then after k steps the residuals satisfy*

$$\|r_k\| \leq \left(1 - \frac{c_A^2}{C_A^2}\right)^{\frac{1}{2}} \|r_{k-1}\|. \tag{5.38}$$

The next theorem follows directly.

Theorem 5.10 *If $c_A > 0$, then after k steps the residuals satisfy*

$$\|r_k\| \leq \left(1 - \frac{c_A^2}{C_A^2}\right)^{\frac{k}{2}} \|r_0\|. \tag{5.39}$$

These two classical results for A may then be applied to the preconditioned system AP^{-1}.

5.2.3 Preconditioned GMRES Theory

By replacing A with AP^{-1}

$$c_{AP^{-1}} = \min_{\|v\|=1} v^T AP^{-1} v$$

$$C_{AP^{-1}} = \max_{\|v\|=1} \|AP^{-1} v\|.$$

The two preceding theorems have analogous results as well.

Theorem 5.11 *If $c_{AP^{-1}} > 0$ then after k steps the residuals satisfy*

$$\|r_k\| \leq \left(1 - \frac{c_{AP^{-1}}^2}{C_{AP^{-1}}^2}\right)^{\frac{1}{2}} \|r_{k-1}\|. \tag{5.40}$$

Theorem 5.12 *If $c_{AP^{-1}} > 0$ then after k steps the residuals satisfy*

$$\|r_k\| \leq \left(1 - \frac{c_{AP^{-1}}^2}{C_{AP^{-1}}^2}\right)^{\frac{k}{2}} \|r_0\|. \tag{5.41}$$

Of course the condition $c_{AP^{-1}} > 0$ follows if AP^{-1} if positive definite, which by Theorem 5.2 it is for sufficiently small H.

We now obtain different bounds for $\|r_k\|$, in terms of $\|AP^{-1} - I\|$, for the classical theory applied to the preconditioned system. First consider $c_{AP^{-1}}$ and $\|v_h\| = 1$. Then

$$\begin{aligned} 1 - v_h^T AP^{-1} v_h &= v_h^T I v_h - v_h^T AP^{-1} v_h \\ &= v_h^T (I - AP^{-1}) v_h \\ &\leq \|v_h\| \, \|I - AP^{-1}\| \, \|v_h\| \\ &= \|I - AP^{-1}\|. \end{aligned}$$

Thus

$$1 - \|I - AP^{-1}\| \leq v_h^T AP^{-1} v_h$$

from which follows

$$c_{AP^{-1}} \geq 1 - \|I - AP^{-1}\|. \tag{5.42}$$

Now consider $C_{AP^{-1}}$ and $\|v_h\| = 1$. Then

$$\begin{aligned} \|AP^{-1}v_h\| - 1 &= \|AP^{-1}v_h\| - \|v_h\| \\ &\leq \|AP^{-1}v_h - v_h\| \\ &\leq \|AP^{-1} - I\| \, \|v_h\|. \end{aligned}$$

Thus

$$\|AP^{-1}v_h\| \leq 1 + \|AP^{-1} - I\|$$

from which it follows that

$$C_{AP^{-1}} \leq 1 + \|AP^{-1} - I\|. \tag{5.43}$$

Combining Equation (5.43) and (5.42) yields

$$\begin{aligned} 1 - \frac{c_{AP^{-1}}^2}{C_{AP^{-1}}^2} &\leq 1 - \frac{(1 - \|AP^{-1} - I\|)^2}{(1 + \|AP^{-1} - I\|)^2} \\ &= \frac{4\|AP^{-1} - I\|}{(1 + \|AP^{-1} - I\|)^2} \\ &\leq 4\|AP^{-1} - I\| \end{aligned}$$

which leads to the following two preconditioned *GMRES* convergence theorems.

Theorem 5.13 *If* $c_{AP^{-1}} > 0$ *then after* k *steps the residuals satisfy*

$$\|r_k\| \leq 4\|AP^{-1} - I\|^{\frac{1}{2}} \|r_{k-1}\|. \tag{5.44}$$

Theorem 5.14 *If* $c_{AP^{-1}} > 0$ *then after* k *steps the residuals satisfy*

$$\|r_k\| \leq 4\|AP^{-1} - I\|^{\frac{k}{2}} \|r_0\|. \tag{5.45}$$

In the above two theorems a stronger requirement than $c_{AP^{-1}} > 0$ is that AP^{-1} be positive definite, which, for sufficiently small H, is true.

Thus we see that if P^{-1} is a close approximation to A^{-1} then *GMRES* will converge quickly. However, we would like to get a more specific estimate on the residuals and appeal to the properties of the preconditioned system and the L-norm. Theorem 5.15 uses the previously developed properties of the preconditioner P and the preconditioned system AP^{-1} to give better bounds on the residuals.

Theorem 5.15 *The residuals of the preconditioned system satisfy*

$$\|r_k\| \leq \mathcal{O}(H)\|r_{k-1}\|. \tag{5.46}$$

Proof: : First note that $q_1(AP^{-1})$ has the form $I - \lambda AP^{-1}$ for some real λ. From Theorem 5.8

$$\begin{aligned} \|r_k\| &\leq \min_{q_1} \|q_1(AP^{-1})\| \, \|r_{k-1}\| \\ &= \min_{\lambda} \|I - \lambda AP^{-1}\| \, \|r_{k-1}\| \\ &\leq \|I - AP^{-1}\| \, \|r_{k-1}\| \\ &\leq \mathcal{O}(H)\|r_{k-1}\| \end{aligned}$$

where we have chosen $\lambda = 1$ and applied Theorem 5.4 on $\|I - AP^{-1}\|$. •

The next result follows directly.

Theorem 5.16 *If the initial residual r_0 is determined using the domain decomposition discretization solve then the residuals of the preconditioned system satisfy*

$$\|r_k\| \leq \mathcal{O}(H^k)\|r_0\| = \mathcal{O}(H^{k+1}). \tag{5.47}$$

Proof: The first inequality is clear by induction. Using $Pu_H = c$ and $Au_h = b$ we know

that $b = c + d$ where $\|d\| = \mathcal{O}(H)$. Then

$$\begin{aligned} \|r_0\| &= \|b - AP^{-1}(Pu_H)\| \\ &= \|c + d - AP^{-1}(Pu_H)\| \\ &\leq \mathcal{O}(H) + \|c - AP^{-1}(Pu_H)\| \\ &= \mathcal{O}(H) + \|Pu_H - Au_H\| \\ &= \mathcal{O}(H) + \|(P - A)u_H\| \\ &= \mathcal{O}(H) \end{aligned}$$

and the result follows. •

5.3 Summary

We have shown that $\|A - P\| = \mathcal{O}(H)$ and that P^{-1} is a good approximation of A^{-1}. For sufficiently small H, AP^{-1}retains the positive definiteness of A. We have extended classical *GMRES* theory to include preconditioned systems, and have obtained convergence rate estimates based on the preconditioned system. Finally, using the properties of our preconditioner, we obtained even sharper convergence rate estimates.

Chapter 6

NUMERICAL RESULTS

In this chapter we present the results on the use of our preconditioner on a suite of four test problems of varying complexity. Test problems were selected so that the exact solution is known. Our preconditioner is applied to the iterative solver *GMRES*. $\mathcal{O}(h^4)$ results are presented for both box and strip decompositions with four grid sizes. The effectiveness of our DD approach using only $\mathcal{O}(h^2)$ accuracy is demonstrated by reporting on another DD preconditioner as proposed by Gropp and Keyes [4]. Parallel results are presented for a shared memory computer with 16 processors.

6.1 The Test Problems

All problems are defined over the unit square and have Dirichlet boundary conditions. Problem 4 is included only for comparison purposes with the Gropp and Keyes [4] $\mathcal{O}(h^2)$ method (*HODIEX* is exact on this problem with $\mathcal{O}(h^4)$ accuracy).

1. The self adjoint variable coefficient PDE

$$\left(e^{xy}u_x\right)_x + \left(e^{-xy}u_y\right)_y - \frac{u}{1+x+y} = g(x,y)$$

   which has the true solution

$$u(x,y) = 0.75e^{xy}\sin(\pi x)\sin(\pi y)\,.$$

2. The constant coefficient PDE

$$u_{xx} + u_{yy} = 6xye^{x+y}(xy + x + y - 3) = g(x,y)$$

which has the true solution

$$u(x,y) = 3e^{x+y}\left(x - x^2\right)\left(y - y^2\right).$$

3. The constant coefficient with cross derivative PDE

$$4u_{xx} - u_{xy} + 4u_{yy} = g(x,y)$$

with true solution

$$u(x,y) = (x - 3y)^2 e^{x-y}.$$

4. The pure isotropic diffusion PDE

$$u_{xx} + u_{yy} = 4$$

where the true solution is

$$u(x,y) = x^2 + y^2.$$

6.2 $\mathcal{O}(h^4)$ Results

Grid resolutions of $h = \frac{1}{24}$, $\frac{1}{48}$, $\frac{1}{96}$, and $\frac{1}{192}$ are reported on. This choice allows domain decompositions in which $\frac{H}{h}$ may be held constant across the different grid sizes.

For each grid size results are presented for a direct solve, an iterative solve with no preconditioning, and an iterative solve with preconditioning for $\frac{H}{h} = 2$, 4, 6, 8, and 12. We present results for both BOXES and vertical STRIPS in which $\frac{H_x}{h_x} = 2$, 4, 6, 8, and 12. The complete set of data for the $\mathcal{O}(h^4)$ *HODIEX* DD results on Problems 1, 2, and 3 are given in Tables A.1 through A.3 in Appendix *A*. These results are presented graphically in Figures 6.1 through 6.12.

Although the smaller grids have negligible execution times, they are reported to enable a study of the rate of convergence in terms of the iteration count. We report the infinity norm error taken over the grid points, the discretization time, the solve time (including both GMRES iterations and the preconditioner factorization), and the number of iterations. The

convergence criteria was to obtain an error with the same order of magnitude as that of the direct solver.

The direct solver was the ELLPACK banded GE with row equilibriation. The only iterative solver used was *GMRES*. Since some problems are self adjoint or have constant coefficients, they could be solved more cheaply with other techniques; however, our interest is to consider DD preconditioning of general linear two dimensional PDEs with variable coefficients, for which *GMRES* has proven to be a robust solver.

For each test problem there is a sequence of three graphs. The first graph plots $Log(GridSize)$ $(GridSize = \frac{1}{h})$ against $Log(Time)$. The next two graphs plot $\frac{H}{h}$ against $Iterations$ and then $Time$ for each grid size. All data in the $\frac{H}{h}$ against $Time$ graph has been normalized to start at $Time = 1$ for $\frac{H}{h} = 2$ for ease of viewing.

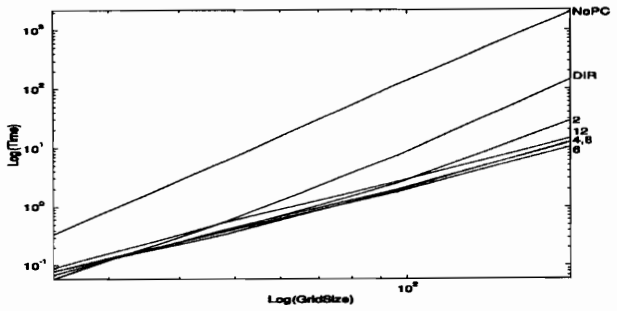


Figure 6.1: Problem 1 Boxes. Grid Size vs Time.

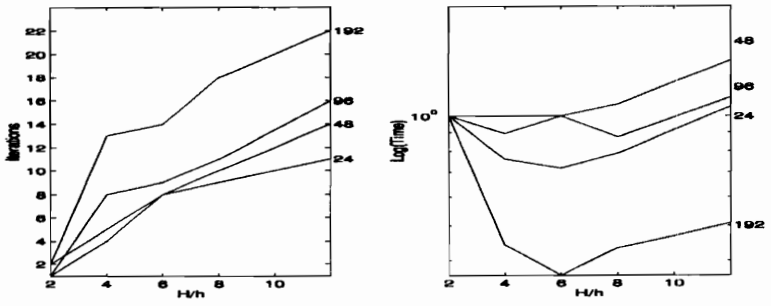


Figure 6.2: Problem 1 Boxes. Iterations and Time vs $\frac{H}{h}$.

Problem 1 BOXES.

Figures 6.1 and 6.2 are Problem 1 with BOXES. Figure 6.1 plots $Log(GridSize)$ against $Log(Time)$. $NoPC$ indicates no preconditioning, DIR indicates the direct solve, and 2, 4, 6, 8, and 12 indicate the corresponding $\frac{H}{h}$. The high execution time for unpreconditioned *GMRES* is typical. BOXES are an order of magnitude faster than the direct solve, and the advantage grows as the grid size increases.

Figure 6.2 shows two plots. The first plot shows $\frac{H}{h}$ against *Iterations* for each grid size. The smaller grid sizes have a lower number of iterations as expected; however, the number of iterations is only slowly increasing for a fixed $\frac{H}{h}$. We see that going from a $\frac{1}{h} = 24$ to 192, a factor of eight, only doubles the number of iterations for a fixed $\frac{H}{h}$. The second plot in Figure 6.2 shows $\frac{H}{h}$ against $Log(Time)$. This plot shows the time decreasing and then increasing as $\frac{H}{h}$ increases. The best execution times are for $\frac{H}{h} \approx 6$.

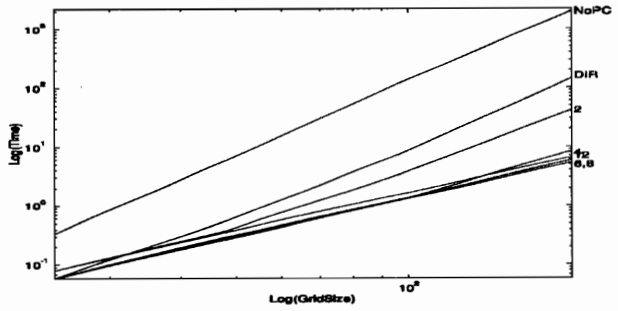


Figure 6.3: Problem 1 STRIPS. Grid Size vs Time.

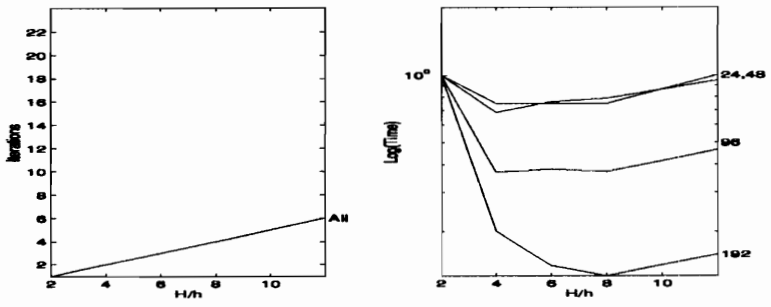


Figure 6.4: Problem 1 STRIPS. Iterations and Time vs $\frac{H}{h}$.

Problem 1 STRIPS.

Figures 6.3 and 6.4 are Problem 1 with STRIPS. Figure 6.3 is similar to Figure 6.1; however, the execution time for STRIPS is better than that for BOXES (except for $\frac{H}{h} = 2$) and is two orders of magnitude faster than the direct solve. Again, we see that the advantage of the DD preconditioned solver over the direct solver grows with grid size.

Figure 6.4 shows two plots. The first plots shows $\frac{H}{h}$ against *Iterations* for each grid size. This time, however, there is only one plot line as all grid sizes have an identical number of iterations. The number of iterations increases linearly as the grid size increases and remains constant for fixed $\frac{H}{h}$. Note that the number of iterations needed to converge is substantially less than it was for BOXES. The second plot in Figure 6.4 shows $\frac{H}{h}$ against $Log(Time)$. As for the BOX decomposition, there is a value which appears to be optimal—this time for $\frac{H}{h} \approx 8$. However, the time for larger values of $\frac{H}{h}$ is not substantially greater.

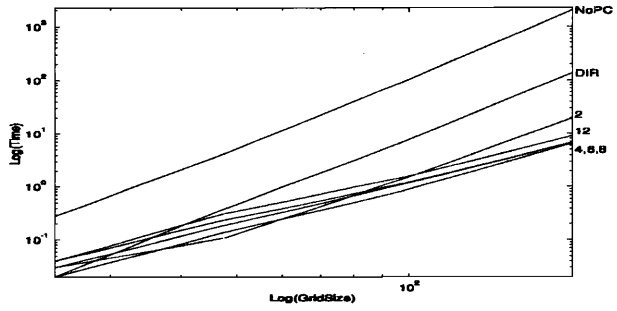


Figure 6.5: Problem 2 BOXES. Grid Size vs Time.

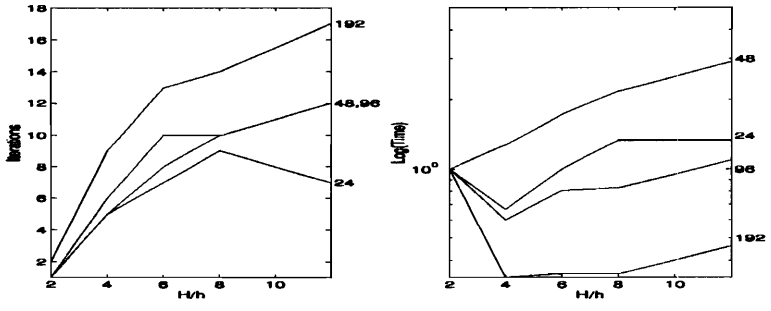


Figure 6.6: Problem 2 BOXES. Iterations and Time vs $\frac{H}{h}$.

Problem 2 BOXES.

Figures 6.5 and 6.6 are for Problem 2 with BOXES. Results are very similar to Problem 1 with BOXES. The execution time for BOXES is again an order of magnitude faster than the direct solve, with $\frac{H}{h} = 2$ performing the worst again. Unpreconditioned *GMRES* again has execution time much higher than the direct solve (as it does on all problems).

Figure 6.6 shows two plots. The first plot shows $\frac{H}{h}$ against *Iterations* for each grid size. The number of iterations is slowly increasing as $\frac{H}{h}$ increases. Again, going from a grid size of $\frac{1}{h} = 24$ to 192, only doubles the number of interations.

The second plot in Figure 6.6 shows again the relative stability in the performance of the method as $\frac{H}{h}$ grows. The best execution times are for $\frac{H}{h} \approx 4$.

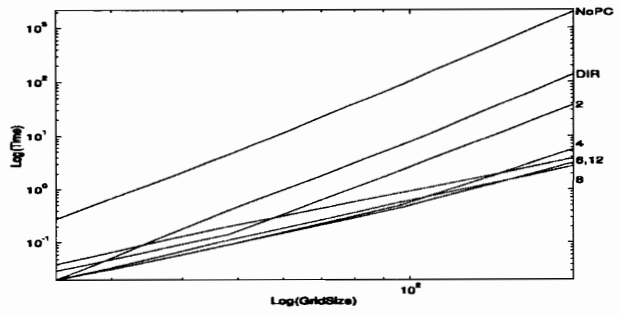


Figure 6.7: Problem 2 STRIPS. Grid Size vs Time.

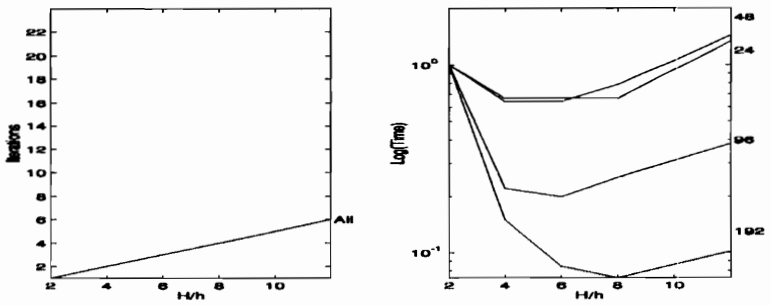


Figure 6.8: Problem 2 STRIPS. Iterations and Time vs $\frac{H}{h}$.

Problem 2 STRIPS.

Figures 6.7 and 6.8 are for Problem 2 with STRIPS. Figure 6.7 is similar to Figure 6.5 for BOXES. However, the execution times for STRIPS is better than that for BOXES and is again two orders of magnitude faster than the direct solve, with the exception being $\frac{H}{h} = 2$, in which STRIPS did not perform as well as BOXES.

Figure 6.8 shows two plots. The first plots shows $\frac{H}{h}$ against *Iterations* for each grid size although there is only line as all plot points coincide. The number of iterations increases linearly as the grid size increases and remains constant for fixed $\frac{H}{h}$.

The second plot of Figure 6.8 shows $\frac{H}{h}$ against $Log(Time)$. This plot shows the time decreasing and then increasing slightly as $\frac{H}{h}$ increases. The best execution times are for $\frac{H}{h} \approx 8$.

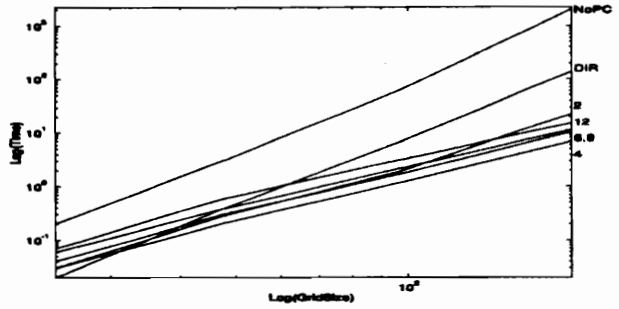


Figure 6.9: Problem 3 BOXES. Grid Size vs Time.

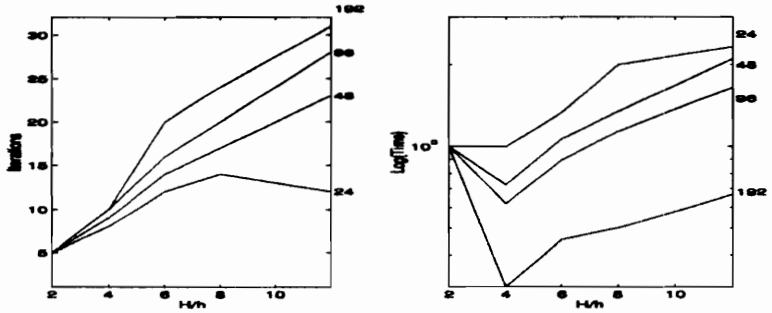


Figure 6.10: Problem 3 BOXES. Iterations and Time vs $\frac{H}{h}$.

Problem 3 BOXES.

Figures 6.9 and 6.10 are for Problem 3 with BOXES. Figure 6.9 plots $Log(GridSize)$ against $Log(Time)$. The execution time for BOXES is again an order of magnitude faster than the direct solve as in the previous cases.

Figure 6.10 shows two plots. The first plots shows $\frac{H}{h}$ against *Iterations* for each grid size. The smaller grid sizes have a lower number of iterations, and for grid size 24 we can see how the curve begins to decrease as $\frac{H}{h}$ increases. The other curves would also show this characteristic if $\frac{H}{h}$ were made large enough. The number of iterations going from the smallest grid size to the largest increases by nearly a factor of 3. For any particular grid size the iteration count is slowly increasing as $\frac{H}{h}$ increases.

The second plot shows $\frac{H}{h}$ against $Log(Time)$. This plot shows the time decreasing and then increasing as $\frac{H}{h}$ increases. The best execution times are for $\frac{H}{h} \approx 4$.

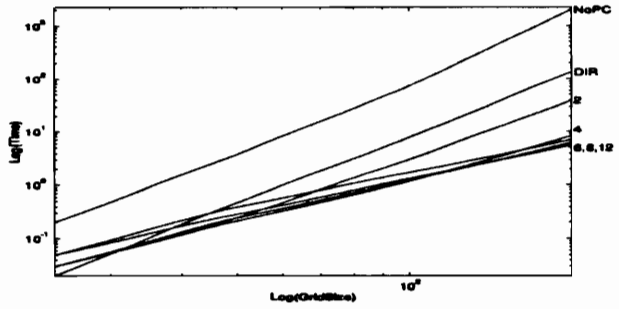


Figure 6.11: Problem 3 STRIPS. Grid Size vs Time.

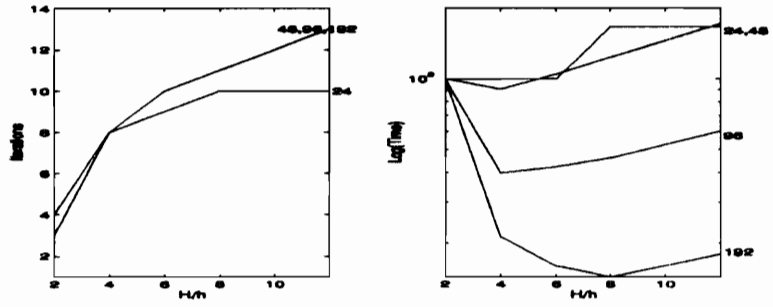


Figure 6.12: Problem 3 STRIPS. Iterations and Time vs $\frac{H}{h}$.

Problem 3 STRIPS.

Figures 6.11 and 6.12 are for Problem 1 with STRIPS. Figure 6.11 is similar to Figure 6.3 and Figure 6.7. The execution time for STRIPS is better than that for BOXES and is two orders of magnitude faster than the direct solve except for $\frac{H}{h} = 2$ (which does not perform well).

Figure 6.12 shows two plots. The first plot shows $\frac{H}{h}$ against *Iterations* for each grid size although in this case all plots do not coincide indicating the preconditioner is not optimal for this problem.

The second plot shows $\frac{H}{h}$ against $Log(Time)$. This plot shows the time decreasing and then increasing as $\frac{H}{h}$ increases. The best execution times vary from $\frac{H}{h} = 4$ to 8.

Table 6.1: Problem 1. $\mathcal{O}(h^2)$.

	HODIEX			Gropp & Keyes		
H/h	Iterations	Time	Factor	Iterations	Time	Factor
Direct	1	285.5	1.0	1	867	1.0
64	4	72.1	4.0	26	353	2.5
32	2	18.5	15.4	32	152	5.7
16	1	5.8	49.2	29	71	12.2
8	1	3.4	82.8	26	57	15.2
4	1	4.7	60.4	21	88	9.9

6.3 $\mathcal{O}(h^2)$ Results

Problems 1 and 4 were selected since they were also reported on by Gropp & Keyes [4] (GK) in their $\mathcal{O}(h^2)$ preconditioned DD method which also used *GMRES* as the iterative solver. This comparison allows us to see the effectiveness of the *HODIEX* DD method when only $\mathcal{O}(h^2)$ results are sufficient. These problems represent the best and worst cases for GK.

There are various ways of discretizing the interfaces. *HODIEX* uses two-dimensional coarse/fine stencils. GK uses tangential interface solves. The tangential approach assumes that a single line segment between two cross points is a subdomain. The solution at the grid points on that line segment are approximated by solving a two-point boundary value problem with boundary values determined by the cross point values or the physical boundary. The discretization is thus a one-dimensional discretization of the PDE that remains when the derivatives normal to the interface are set to zero. With this technique, P does not represent a coefficient matrix of a lessor order discretization (as it does with our *HODIEX* DD preconditioner).

The *HODIEX* code is written in Fortran 77 and the GK code is written in C and the two codes are executed on different processors. Different organizations of code and compiler capabilities across different computer architectures lead to variations in execution times, so that the absolute execution times are not meaningful. However, the speedup of the iterative

Table 6.2: Problem 4. $\mathcal{O}(h^2)$.

	HODIEX			Gropp & Keyes		
H/h	Iterations	Time	Factor	Iterations	Time	Factor
Direct	1	288.9	1.0	1	865	1.0
64	1	62.2	4.6	10	249	3.5
32	1	16.5	17.6	11	80	10.8
16	1	6.0	48.2	9	27	32.0
8	1	3.5	85.0	7	14	61.8
4	1	4.8	61.5	6	21	41.2

algorithm relative to the direct solve time on a particular computer is an indication of the performance of an algorithm. Also, the iteration count is dependent not on computer architectures as much as on the algorithms involved.

Tables 6.1 and 6.2 compare a *HODIEX* $\mathcal{O}(h^2)$ discretization and the *HODIEX* DD preconditioner with the $\mathcal{O}(h^2)$ GK results for $\frac{1}{h} = 128$. Time represents the sum of both preconditioner factorization, application, and the *GMRES* iterations. Factor is the ratio of direct time to iterative time.

Convergence criteria for both schemes was the reduction of the residual to 10^{-5}. As $\frac{H}{h}$ decreases (more subdomains), the *HODIEX* iteration count always decreases (until equal to 1), while GK shows an initial increase in iterations followed by a decrease. However, both *HODIEX* and GK have optimal execution times at $\frac{H}{h} = 8$ and the general pattern of reduction in execution times is similar, although *HODIEX* achieves a much better reduction in solve time.

HODIEX always requires fewer iterations than GK (only 1 for $\frac{H}{h} < 32$). Tables 6.1 and 6.2 show that when only $\mathcal{O}(h^2)$ accuracy is required, DD preconditioned *HODIEX* is essentially a direct solver and only one iteration of *GMRES* is required. Also, *HODIEX* performs equally well on both problems while GK performs poorly on Problem 1 as compared to Problem 2.

There are two reasons that *HODIEX* performs so much better than GK. First, the

HODIEX preconditioner uses a two-dimensional stencil for all grid points (interior points use a fine/fine discretization, interface points use a fine/coarse discretization, and the cross-point system is a coarse/coarse discretization). The GK preconditioner uses one-dimensional interface system approximations. Thus the *HODIEX* preconditioner P is a better approximation of A. Second, since the *HODIEX* preconditioner actually solves the PDE, a good initial guess becomes available for no extra computational cost.

6.4 Parallel Results

Tables 6.3 and 6.4 present computation times on a Sequent Symmetry S81[1] for various numbers of processors, along with S_p = (sequential time)/(time for p processors) and efficiency $\omega = S_p/p$. *PreCond Time* is the time for the preconditioner factorization and application. *GMRES Time* is the time for the *GMRES* iterations. Speedup and efficiency are based on the *Total Time*.

The data is the average of three runs for Problem 1 with a 4×4 and a 16×1 DD in which $\frac{1}{h} = 96$. These decompositions were selected in an attempt to give all processors maximum work on the interior solves since they resulted in 16 interior subdomains and we used up to 16 processors for the tests.

The following modules were parallelized:

DAXPY Constant times a vector plus a vector.

DDOT Dot product of two vectors.

DCOPY Copy one vector to another.

MV Sparse matrix vector multiply.

PF Preconditioner factorization.

PS Preconditioner solve.

[1]Argonne Advanced Computing Research Facility

Table 6.3: Problem 1. Parallel 4 × 4 DD.

p	PreCond Time	GMRES Time	Total Time	S_p	ω
1	50.15	175.95	226.10	-	-
2	39.85	90.12	129.97	1.7	87
4	18.42	47.68	66.10	3.4	85
8	9.23	26.67	35.90	6.3	79
16	3.35	16.32	19.67	11.5	72

Table 6.4: Problem 1. Parallel 16 × 1 DD.

p	PreCond Time	GMRES Time	Total Time	S_p	ω
1	7.77	11.70	19.47	-	-
2	5.63	6.97	12.60	1.5	75
4	4.70	4.60	9.30	2.1	53
8	4.22	3.40	7.62	2.6	33
16	3.97	2.85	6.82	2.9	18

GMRES Generalized minimum residual iterative solver.

Except for *GMRES*, the modules listed above are relatively simple with only one or two loops and all loops were parallelized. In *GMRES* seven loops were parallelized and only one loop was not parallelized. The unparallelized loop involved a transformation of the Hessenburg system. The modified Gram-Schmidt loop was not parallelized but it contains parallelized *DAXPY* and *DDOT* operations. Also, only the inner loop of the upper triangular backsolve was parallelized. Each iteration of *GMRES* contains both *PF* and *PS* which were parallelized.

Table 6.3 shows good parallel performance with a speedup of 11.5 on 16 processors for the 4 × 4 BOXES. Table 6.4 shows poor performance for the 16 × 1 STRIPS. These performances are made even more apparent in the plots shown in Figures 6.13 and 6.14.

One area that the BOX performance can be improved is in the preconditioner solve. In

this step there are four loops for the interior, x-interface, y-interface, and crosspoint subdomains which are processed sequentially. The x-interface and y-interface solves could be done in parallel with more sophisticated programming. Also note that there are only 4 y-interface and 4 x-interface subdomains, so more than 4 processors does not improve performance. A larger DD with more interface subdomains would achieve better performance.

The interface solves effect the 16×1 STRIP performance even more. In this case there is only 1 interface subdomain and thus the interface solve is essentially a sequential operation. Instead of parallelizing the outer loop, an inner loop parallelization would probably be more effective on STRIPS. Another reason STRIP parallel performance is not good is that STRIPS sequential time is much smaller than BOX sequential time (19 for STRIPS and 226 for BOXES).

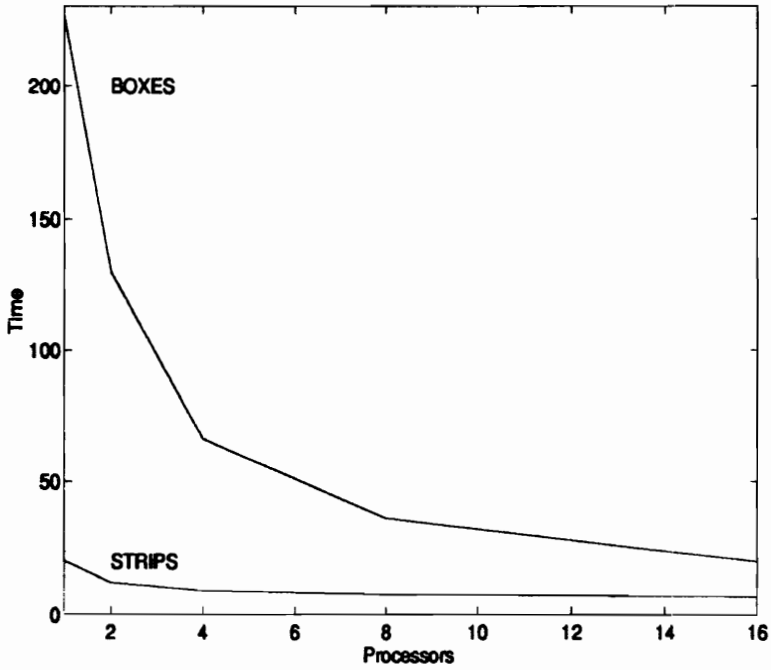


Figure 6.13: Problem 1 Parallel. Processors vs Time.

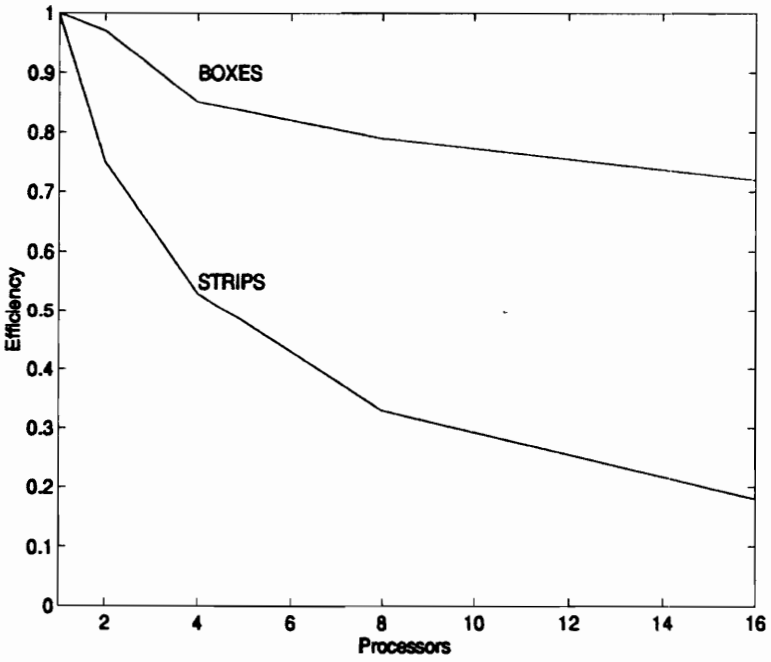


Figure 6.14: Problem 1 Parallel. Processors vs Efficiency.

6.5 Summary

Except for the smallest grid sizes (whose solve times are small), the discretization times to compute A and P are small when compared to the solve time. For both BOXES and STRIPS the iteration count continues to decrease as the number of subdomains increases. This is to be expected since the preconditioner is approaching the coefficient matrix. However, the optimal execution times for both BOXES and STRIPS is at $\frac{H}{h} = 6$ or $\frac{H}{h} = 8$, even though more subdomains results in fewer iterations. This is due to the fact that as the number of subdomains grow, the interior subdomain solves get smaller while the number of interior solves get larger.

The preconditioner performs well, especially on STRIPS, which consistently outperformed boxes. The $\frac{H}{h}$ ratio which gives best performance is the same for both STRIPS and BOXES. Presenting the results by fixed H/h makes it easier to estimate the optimality of the preconditioner. An optimal preconditioner is one in which a fixed value of H/h has the same number of iterations as the grid increases, and the number of iterations to solve the preconditioned system to a given accuracy is independent of h. When H/h is held constant for BOXES, the iteration count increases slowly indicating a nearly optimal preconditioner. For STRIPS, when H/h is held constant then the iteration count is constant indicating an optimal preconditioner.

There are several possible reasons why strips perform better than boxes for a fixed $\frac{H}{h}$. First, there is no crosspoint subdomain and hence no inaccurate coarse grid solve for those grid points. Second, the interface solve for the other grid points is more accurate because it now uses true boundary values rather than the inaccurate estimates given by the coarse grid solve. Third, there are fewer subdomains to solve resulting in more interior points per subdomain (larger interior solves) with correspondingly more accurate solves. These all result in a more accurate initial guess and a better preconditioner; and thus fewer iterations and execution time for GMRES.

In all cases the unpreconditioned solve times were substantially greater than the direct

solve time. However, the iterative solve time was substantially better than the direct solve time and the larger the grid the greater the factor. The $\mathcal{O}(h^2)$ results show that *HODIEX* DD is also a good direct solver if only limited accuracy is required.

My research has shown that coefficient matrices resulting from discretization of the PDE make good preconditioners. Domain decomposition makes the cost of obtaining and applying such preconditioners relatively inexpensive with the additional advantage of providing for a good "initial guess" for the iterative solver.

Chapter 7

CONCLUSIONS

7.1 Summary

Out goal was to investigate the use of higher order discretizations with a preconditioned iterative solver. We first described a high order discretization scheme *HODIEX* which uses compact 3×3 stencils to discretize a general second order linear PDE with variable coefficients. We discussed the construction and solution of the *HODIEX* equations. Using similar solution techniques we derived an alternative 9-point compact stencil whose desirable properties have been proved by many authors in the past.

To illustrate the achieved order of accuracy of our method we considered *HODIEX* on a suite of test problems. Each test problem was run using a direct solver and the order of accuracy was computed for the various cases. By considering *HODIEX* exact on several linear spaces of increasing dimension, we showed not only that each problem did achieve a high order of accuracy, but also that some problems had an order of accuracy as high as $\mathcal{O}(h^8)$. It was shown that problems with cross derivatives, although achieving a higher order accuracy than standard differences, did not achieve as high an order as problems with no cross derivative. We also saw that for a given accuracy *HODIEX* was much faster than standard second order finite differences.

We proved that *HODIEX* generates stencils that are $\mathcal{O}(h)$ perturbations of standard finite difference stencils. Since it has been proven that the standard finite difference discretizations converge, then for sufficiently small h so does *HODIEX*. Similarly, the linear system resulting from a *HODIEX* discretization has the same desirable properties for sufficiently small h. These properties include diagonal dominance, definiteness, monotonicity,

and irreducibility. Other properties that linear systems generated by *HODIEX* share with those generated by standard finite differences include bandedness, sparseness, and block tridiagonal form. With these properties both direct and iterative solvers successful for the low order method can be applied with *HODIEX*.

We saw that existence of the *HODIEX* discretization was related to the null space of the PDE operator, and that there was a maximum possible order for a given operator. We saw that a *HODIEX* discretization exact on $\mathcal{P}_k$ had a truncation error $\mathcal{O}(h^{k-1})$ and thus a *HODIEX* discretization for $k \geq 2$ was consistent. We showed that if a non-trivial *HODIEX* discretization exists then the truncation error and the discretization error are of the same order.

We described a domain decomposition technique for constructing a DD approximation of the linear system which not only was a good preconditioner, but also gave a good initial estimate for the iterative solver. We described our DD algorithm for solving the PDE and also viewed this algorithm in terms of the matrix structure. We discussed parallelization issues using our *HODIEX* DD preconditioner.

We showed the preconditioner is an $\mathcal{O}(H)$ perturbation of the linear system generated by standard finite differences and proved a number of properties of the preconditioned linear system. We discussed the Krylov subspace solver *GMRES* and proved that the properties of our preconditioner give sharper convergence rate estimates. These estimates were given first in terms of the classical theory and then in terms of a new preconditioned *GMRES* theory.

Finally we considered a suite of problems to test our *HODIEX* DD preconditioned *GMRES* method. We presented results on a number of grid sizes for various box and strip domain decompositions. These results demonstrated that our method works well with rapid convergence and high accuracy. Since out method is variable order, we compared it with a well known $\mathcal{O}(h^2)$ method. We discovered that not only did *HODIEX* DD outperform the other method, but that *HODIEX* is actually a very fast direct solver.

We presented parallel results on one problem utilizing a shared memory multiprocessor

with up to 16 processors. The speedups obtained were good for BOXES and not good for STRIPS. The reasons for this were discussed along with suggestions for improving parallel performance.

7.2 Future Work

The research in this dissertation has uncovered many areas for future study. We only considered one preconditioner; however, other preconditions for a PDE discretized with *HODIEX* are possible. PDE's with non-Dirichlet boundary conditions need to be considered. Parallel performance on other parallel processors, such as distributed memory machines, needs to be investigated.

We have concentrated on 9 point compact stencils. Small stencils such as five point star also have *HODIEX* discretizations, as well as larger stencils such as 25 point compact stencils and even larger non-compact stencils. While we only considered second order PDEs, *HODIEX* may also be extended to higher order differential equations, and to equations in more than two dimensions. The *HODIEX* method could also be applied to other types of equations such as parabolic or hyperbolic.

Domains and subdomains of various shapes and subdomains with different numbers of grid points should be studied. *HODIEX* discretizations exact on non-polynomial basis functions can also be considered. *HODIEX* discretizations on non-uniform grids also need to be studied.

We only considered one iterative solver; other solvers need to be studied. The main reason our preconditioner works so well is that it is two-dimensional. This approach can be considered with other discretization techniques such as finite elements. *HODIE* is not limited to second order operators, elliptic operators, or two dimensional PDEs and may even be applied to ODEs.

At this point there is little theory on why strips perform so well. Most other DD methods reported on in the literature show poor performance for strips (or at least no better). A better understanding of why strips perform well might lead to other good preconditioners.

Appendix A

Table A.1: Problem 1. $(e^{xy}u_x)_x + (e^{-xy}u_y)_y - \frac{u}{1+x+y} = g(x,y)$.

	BOXES				STRIPS			
GRID	24x24	48x48	96x96	192x192	24x24	48x48	96x96	192x192
	Direct Solve							
err	.21D-5	.15D-6	.10D-7	.69D-9				
dis	.04	.17	.71	3.11				
sol	.02	.38	6.59	137.35				
	No Preconditioning							
its	132	476	2070	2500[1]	132	476	2070	2500[1]
err	.21D-5	.15D-6	.10D-7	.96D-3	.21D-5	.15D-6	.10D-7	.96D-3
dis	.04	.17	.71	3.11	.04	.17	.71	3.11
sol	.30	5.74	116.09	649.32	.30	5.74	116.09	649.32
	$H/h = 12$				$H_x/h_x = 12$			
DD	2x2	4x4	8x8	16x16	2x1	4x1	8x1	16x1
its	11	14	16	22	6	6	6	6
err	.20D-5	.13D-6	.10D-7	.47D-9	.20D-5	.12D-6	.12D-7	.72D-9
dis	.04	.19	.79	3.24	.04	.18	.75	3.09
sol	.05	.35	1.84	10.90	.04	.18	.78	3.43
	$H/h = 8$				$H_x/h_x = 8$			
DD	3x3	6x6	12x12	24x24	3x1	6x1	12x1	24x1
its	9	10	11	18	4	4	4	4
err	.19D-5	.14D-6	.10D-7	.50D-9	.18D-5	.18D-6	.11D-7	.70D-9
dis	.04	.20	.85	3.50	.04	.19	.79	3.21
sol	.03	.21	1.11	8.60	.02	.11	.43	2.04
	$H/h = 6$				$H_x/h_x = 6$			
DD	4x4	8x8	16x16	32x32	4x1	8x1	16x1	32x1
its	8	8	9	14	3	3	3	3
err	.21-5	.15D-6	.99D-8	.54D-9	.22D-5	.14D-6	.96D-8	.58D-9
dis	.05	.22	.89	3.70	.05	.20	.80	3.30
sol	.03	.16	.89	6.49	.01	.09	.45	2.50
	$H/h = 4$				$H_x/h_x = 4$			
DD	6x6	12x12	24x24	48x48	6x1	12x1	24x1	48x1
its	5	4	8	13	2	2	2	2
err	.17D-5	.89D-7	.99D-8	.67D-9	.23D-5	.15D-6	.10D-7	.70D-9
dis	.06	.25	.99	4.08	.05	.21	.86	3.54
sol	.02	.09	.90	8.27	.01	.05	.35	4.77
	$H/h = 2$				$H_x/h_x = 2$			
DD	12x12	24x24	48x48	96x96	12x1	24x1	48x1	96x1
its	2	1	1	2	1	1	1	1
err	.19D-5	.73D-7	.41D-8	.57D-9	.19D-5	.14D-6	.94D-8	.59D-9
dis	.07	.29	1.20	5.35	.06	.25	1.03	4.46
sol	.01	.09	1.28	22.54	.02	.13	2.26	36.83

[1] = Maximum Iterations

Table A.2: Problem 2. $u_{xx} + u_{yy} = 6xye^{x+y}(xy + x + y - 3) = g(x, y)$.

	BOXES				STRIPS			
GRID	24x24	48x48	96x96	192x192	24x1	48x1	96x1	192X1
	Direct Solve							
err	.11D-5	.70D-7	.42D-8	.43D-9				
dis	.00	.02	.06	.45				
sol	.02	.38	6.49	136.13				
	No Preconditioning							
its	98	379	1544	2500[1]				
err	.11D-5	.70D-7	.41D-8	.17D-3				
dis	.00	.02	.08	.47				
sol	.28	4.37	87.27	552.98				
	$H/h = 12$				$H_x/h_x = 12$			
DD	2x2	4x4	8x8	16x16	2x1	4x1	8x1	16x1
its	7	12	12	17	6	6	6	6
err	.12D-5	.99D-7	.64D-8	.35D-9	.11D-5	.80D-7	.42D-8	.42D-9
dis	.00	.02	.08	.32	.00	.02	.08	.30
sol	.04	.30	1.37	8.63	.04	.18	.78	3.43
	$H/h = 8$				$H_x/h_x = 8$			
DD	3x3	6x6	12x12	24x24	3x1	6x1	12x1	24x1
its	9	10	10	14	4	4	4	4
err	.12D-5	.73D-7	.56D-8	.53D-9	.13D-5	.72D-7	.40D-8	.40D-9
dis	.00	.02	.09	.34	.00	.01	.08	.32
sol	.04	.22	1.01	6.45	.02	.10	.49	2.39
	$H/h = 6$				$H_x/h_x = 6$			
DD	4x4	8x8	16x16	32x32	4x1	8x1	16x1	32x1
its	7	8	10	13	3	3	3	3
err	.15D-5	.71D-7	.34D-8	.18D-9	.15D-5	.67D-7	.39D-8	.25D-9
dis	.01	.02	.08	.34	.01	.02	.08	.34
sol	.02	.17	.99	6.44	.01	.07	.37	2.77
	$H/h = 4$				$H_x/h_x = 4$			
DD	6x6	12x12	24x24	48x48	6x1	12x1	24x1	48x1
its	5	5	6	9	2	2	2	2
err	.10D-5	.57D-7	.34D-8	.33D-9	.10D-5	.67D-7	.40D-8	.26D-9
dis	.00	.03	.09	.38	.01	.02	.08	.35
sol	.02	.11	.71	6.17	.01	.07	.42	5.17
	$H/h = 2$				$H_x/h_x = 2$			
DD	12x12	24x24	48x48	96x96	12x1	24x1	48x1	96x1
its	1	1	1	2	1	1	1	1
err	.13D-5	.87D-7	.48D-8	.70D-9	.11D-5	.71D-7	.42D-8	.94D-9
dis	.01	.03	.12	.52	.01	.02	.10	.46
sol	.02	.08	1.21	18.92	.02	.12	2.16	36.40

[1] = Maximum Iterations

Table A.3: Problem 3. $4u_{xx} - u_{xy} + 4u_{yy} = g(x, y)$.

	BOXES				STRIPS			
GRID	24x24	48x48	96x96	192x192	24x1	48x1	96x1	192X1
	Direct Solve							
err	.41D-5	.50D-6	.63D-7	.21D-7				
dis	.00	.02	.06	.45				
sol	.02	.38	6.49	136.13				
	No Preconditioning							
its	88	292		2000+				
err	.38D-5	.49D-6		.82D-3				
dis	.00	.02		.29				
sol	.20	3.17		551.60				
	$H/h = 12$				$H_x/h_x = 12$			
DD	2x2	4x4	8x8	16x16	2x1	4x1	8x1	16x1
its	12	23	28	31	10	13	13	13
err	.41D-5	.50D-6	.63D-7	.23D-7	.41D-5	.56D-6	.60D-7	.22D-7
dis	.01	.02	.07	.33	.00	.02	.07	.30
sol	.06	.59	3.00	14.65	.05	.34	1.50	6.65
	$H/h = 8$				$H_x/h_x = 8$			
DD	3x3	6x6	12x12	24x24	3x1	6x1	12x1	24x1
its	14	17	20	24	10	11	11	11
err	.41D-5	.50D-6	.64D-7	.22D-7	.41D-5	.49D-6	.61D-7	.22D-7
dis	.01	.02	.09	.33	.00	.02	.08	.33
sol	.05	.37	2.03	10.97	.05	.24	1.12	5.25
	$H/h = 6$				$H_x/h_x = 6$			
DD	4x4	8x8	16x16	32x32	4x1	8x1	16x1	32x1
its	12	14	16	20	9	10	10	10
err	.41D-5	.51D-6	.65D-7	.22D-7	.41D-5	.52D-6	.64D-7	.22D-7
dis	.00	.02	.09	.35	.00	.02	.08	.34
sol	.04	.29	1.57	9.87	.03	.20	1.02	5.85
	$H/h = 4$				$H_x/h_x = 4$			
DD	6x6	12x12	24x24	48x48	6x1	12x1	24x1	48x1
its	8	9	10	10	8	8	8	8
err	.34D-5	.42D-6	.50D-7	.22D-7	.41D-5	.50D-6	.64D-7	.22D-7
dis	.00	.02	.09	.38	.00	.02	.08	.33
sol	.03	.19	1.06	6.44	.03	.17	.96	7.86
	$H/h = 2$				$H_x/h_x = 2$			
DD	12x12	24x24	48x48	96x96	12x1	24x1	48x1	96x1
its	5	5	5	5	3	3	3	4
err	.41D-5	.50D-6	.62D-7	.22D-7	.36D-5	.37D-6	.29D-7	.22D-7
dis	.01	.02	.11	.47	.01	.03	.10	.40
sol	.02	.17	1.75	22.02	.02	.18	2.51	38.13

[1] = Maximum Iterations

REFERENCES

[1] W. Arnoldi. "The Principle of Minimized Iterations in the Solution of the Matrix Eigenvalue Problem". *Quarterly Journal of Applied Mathematics*, 9:17–29, 1951.

[2] A. K. Aziz, R. B. Kellog, and A. B. Stephens. "Least Squares Methods for Elliptic Systems". *Mathematics of Computation*, 44(169):53–70, 1985.

[3] A. J. Baker. *Finite Element Computational Fluid Mechanics.* Hemisphere Publishing, New York, NY, 1983.

[4] R. Barrett, M. Berry, and T. Chan. *TEMPLATES for the Solution of Linear Systems: Building Blocks for Iterative Methods.* SIAM, Philadelphia, PA, 1994.

[5] W. G. Bickley. "Finite Difference Formulas for the Square Lattice". *Quarterly Journal of Applied Mathematics*, 1:35–42, 1948.

[6] G. Birkhoff, E. C. Gartland, and R. E. Lynch. "Difference Methods for Solving Convection–Diffusion Equations". *Computing Mathematics Applications*, 19(11):147–160, 1990.

[7] Garrett Birkhoff and Surender Gulati. "Optimal Few Point Discretizations of Linear Source Problems". *SIAM Journal of Numerical Analysis*, 11(4):700–728, 1974.

[8] Garrett Birkhoff and Robert E. Lynch. *Numerical Solution of Elliptic Problems.* SIAM, Philadelphia, PA, 1984.

[9] Garrett Birkhoff and Robert E. Lynch. "Some Current Questions on Solving Linear Elliptic Problems". *Numerical Mathematik*, 57:527–546, 1990.

[10] Garrett Birkhoff and Arthur Schoenstadt, editors. *Elliptic Problem Solvers II.* Academic Press, New York, NY, 1984.

[11] Peter Bjórstad. "Fast Numerical Solution of the Biharmonic Dirichlet Problem on Rectangles". *SIAM Journal of Numerical Analysis*, 20(1):59–71, 1983.

[12] Petter E. Bjórstad and O. B. Widlund. "Iterative Methods for the Solution of Elliptic Problems on Regions Divided into Substructures". *SIAM Journal of Numerical Analysis*, 23(6):1097–1120, 1986.

[13] Petter E. Bjórstad and O. B. Widlund. "To Overlap or Not to Overlap: A Note on a Domain Decomposition Method for Elliptic Problems". *SIAM Journal of Scientific and Statistical Computing*, 10(5):1053–1061, 1989.

[14] Ronald F. Boisvert. "Families of High Order Accurate Discretizations of Some Elliptic Problems". *SIAM Journal of Scientific and Statistical Computing*, 2(3):268–284, 1981.

[15] Ronald F. Boisvert. "High Order Compact Difference Formulas for Elliptic Problems with Mixed Boundary Conditions". In *Advances in Computer Methods for Partial Differential Equations*, pages 193–199, IMACS, New Brunswick, NY, 1981.

[16] Ronald F. Boisvert. *High Order Finite Difference Methods for Elliptic Boundary Value Problems*. PhD Thesis, Purdue University, 1979.

[17] James H. Bramble, R. E. Ewing, Josepj E. Pasciak, and Alfred H. Schatz. "A Preconditioning Technique for the Efficient Solution of Problems with Local Grid Refinement". *Computing Methods in Applied Mechanics and Engineering*, 67:149–159, 1988.

[18] James H. Bramble, Richard E. Ewing, Rossen R. Parashkekov, and Joseph E. Pasciak. "Domain Deomposition Methods for Problems with Partial Refinement". *SIAM Journal of Scientific and Statistical Computing*, 13(11):397–410, 1992.

[19] James H. Bramble and Joseph E. Pasciak. "A Domain Decomposition Technique for Stokes Problems". *Applied Numerical Mathematics*, 6:251–261, 1989/1990.

[20] James H. Bramble and Joseph E. Pasciak. "A Preconditioning Technique for Indefinite Systems Resulting from Mixed Approximations of Elliptic Problems". *Mathematics of Computation*, 50(181):1–17, 1988.

[21] James H. Bramble and Joseph E. Pasciak. "New Convergence Estimates for Multrigrid Algorithms". *Mathematics of Computation*, 49(180):311–329, 1987.

[22] James H. Bramble, Joseph E. Pasciak, and A. H. Schatz. "An Iterative Method for Elliptic Problems on Regions Partitioned into Substructures". *Mathematics of Computation*, 46(174):361–369, 1986.

[23] James H. Bramble, Joseph E. Pasciak, and Alfred H. Schatz. "The Construction of Preconditioners for Elliptic Problems by Substructuring, II". *Mathematics of Computation*, 49(179):1–16, 1987.

[24] James H. Bramble, Joseph E. Pasciak, and Alfred H. Schatz. "The Construction of Preconditioners for Elliptic Problems by Substructuring, III". *Mathematics of Computation*, 51(184):415–430, 1988.

[25] James H. Bramble, Joseph E. Pasciak, and Alfred H. Schatz. "The Construction of Preconditioners for Elliptic Problems by Substructuring, IV". *Mathematics of Computation*, 53(187):1–24, 1989.

[26] James H. Bramble, Joseph E. Pasciak, and Alfred H. Schatz. "The Construction of Preconditioners for Elliptic Problems by Substructuring, I". *Mathematics of Computation*, 47(175):103–134, 1986.

[27] James H. Bramble, Joseph E. Pasciak, Junping Wang, and Jinchao Xu. "Converegence Estimates for Product Iterative Methods with Applications to Domain Decomposition". *Mathematics of Computation*, 57(195):1–21, 1991.

[28] James H. Bramble, Joseph E. Pasciak, Junping Wang, and Jinchao Xu. "Convergence Estimates for Multigrid Algorithms Without Regularity Assumptions". *Mathematics of Computation*, 57(195):23–45, 1991.

[29] James H. Bramble, Joseph E. Pasciak, and Jinchao Xu. "Parallel Multilevel Preconditioning". *Mathematics of Computation*, 55(191):1–22, 1990.

[30] James H. Bramble, Joseph E. Pasciak, and Jinchao Xu. "The Analysis of Multigrid Algorithms for Nonsymmetric and Indefinite Elliptic Problems". *Mathematics of Computation*, 51(184):389–414, 1988.

[31] M. Braun. *Differential Equations and Their Applications*. Springer–Verlag, New York, NY, 1986.

[32] Peter N. Brown and Youcef Saad. "Hybrid Krylov Methods for Nonlinear Systems of Equations". *SIAM Journal of Scientific and Statistical Computing*, 11(3):450–481, 1990.

[33] K. P. Bube and J. C. Strikwerda. "Interior Regularity Estimates for Elliptic Systems of Difference Equations". *SIAM Journal of Numerical Analysis*, 20(4):653–670, 1983.

[34] B. L. Buzbee, G. H. Golub, and C. W. Nielson. "On Direct Methods for Solving Poisson's Equations". *SIAM Journal of Numerical Analysis*, 7(4):627–655, 1970.

[35] Xiao-CHuan Cai, William D. Gropp, and David Keyes. *"Convergence Rate Estimate for a Domain Decomposition Method"*. Math and CS Div. MCS–P202–1290, Argonne National Laboratory, 1991.

[36] Xiao-CHuan Cai and Olof B. Widlund. "Domain Decomposition Algorithms for Indefinite Elliptic Problems". *SIAM Journal of Scientific and Statistical Computing*, 13(1):243–258, 1992.

[37] Tony Chan. "Analysis of Preconditioners for Domain Decomposition". *SIAM Journal of Numerical Analysis*, 24(2):382–390, 1987.

[38] Tony Chan, Weinan E, and Jiachang Sun. "Domain Decomposition Interface Preconditioners for Fourth Order Elliptic Problems". *Applied Numerical Mathematics*, 8:317–331, 1991.

[39] Tony Chan and Danny Goovaerts. "A Note on the Efficiency of Domain Decomposition Incomplete Factorizations". *SIAM Journal of Scientific and Statistical Computing*, 11(4):794–803, 1990.

[40] Tony Chan and Danny Goovaerts. *"Schwarz=Schur: Overlapping Versus Nonoverlapping Domain Decomposition"*. CAM Report 88–21, University of Califormia, 1988.

[41] Tony F. Chan and Faisal Saied. "A Comparison of Elliptic Solvers for General Two-Dimensional Regions". *SIAM Journal of Scientific and Statistical Computing*, 6(3):742–760, 1985.

[42] L. Collatz. *The Numerical Treatment of Differential Equations*. Springer-Verlag, Berlin, 1935.

[43] P. Concus, G. H. Golub, and G. Meurant. "Block Preconditioning for the Conjugate Gradient Method". *SIAM Journal of Scientific and Statistical Computing*, 6(1):220–252, 1985.

[44] Michael G. Crandall and Andrew Majda. "Monotone Difference Approximations for Scalar Conservation Laws". *Mathematics of Computation*, 34(149):1–21, 1980.

[45] Clint Dawson, Qiang Du, and Todd F. Dupont. *"A Finite Difference Domain Decomposition Algorithm for Numerical Solution of the Heat Equation"*. CS Department TR 89–09, University of Chicago, November 1989.

[46] Eusebius J. Doedel. "The Construction of Finite Difference Approximations to Ordinary Differential Equations". *SIAM Journal of Numerical Analysis*, 15(3):450–465, 1978.

[47] Milo R. Dorr. "A Domain Decomposition Preconditioner with Reduced Rank Interdomain Coupling". *Applied Numerical Mathematics*, 8:333–352, 1991.

[48] Craig C. Douglas. "A Tupleware Approach to Domain Decomposition Methods". *Applied Numerical Mathematics*, 8:353–373, 1991.

[49] Jim Douglas and Jean Roberts. "Global Estimates for Mixed Methods of Second Order Elliptic Equations". *Mathematics of Computation*, 44(169):39–52, 1985.

[50] M. Dryja and W. Proskurowski. "Composite Iterative Method for Elliptic Problems in Irregular Regions". *Applied Numerical Mathematics*, 6:263–279, 1989/1990.

[51] M. Dryja and O. B. Widlund. "Some Domain Decomposition Algoritms for Elliptic Problems". In *Iterative Methods for Large Linear Systems*, Academic Press, San Diego, CA, 1989.

[52] M. Dryja and O. B. Widlund. "Towards a Unified Theory of Domain Decomposition Algoritms for Elliptic Problems". In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, SIAM, Philadelphia, PA, 1990.

[53] Todd Dupont, Richard Kendal, and H. H. Rachford. "An Approximate Factorization Procedure for Solving Self-Adjoint Elliptic Difference Equations ". *SIAM Journal of Numerical Analysis*, 5(3):559–573, 1968.

[54] Todd Dupont and Ridgway Scott. "Polynomial Approximation of Functions in Sobolev Spaces". *Mathematics of Computation*, 34(150):441–463, 1980.

[55] Ricardo G. Durán. "On Polynomial Approximation in Sobolev Spaces". *SIAM Journal of Numerical Analysis*, 20(5):985–988, 1983.

[56] Wayne R. Dyksen, Elias N. Houstis, and John R. Rice. "The PDE Population". In John R. Rice and Ronald F. Boisvert, editors, *Solving Elliptic Problems Using ELLPACK*, Springer-Verlag, New York, 1984.

[57] S. Eisenstat, H. Elman, and M. Schultz. "Variational Iterative Methods for Nonsymmetric Systems of Linear Equations". *SIAM Journal of Numerical Analysis*, 20:345–357, 1983.

[58] Howard C. Elman, Youcef Saad, and Paul E. Saylor. *"A Hybrid Chebyshev Krylov Subspace Algoritm for Solving Nonsymmetric Systems of Linear Equations"*. DCS TR 301, Yale University Research Report, 1984.

[59] Bengt Fornberg. "Generation of Finite Difference Formulas on Arbitrarily Spaced Grids". *Mathematics of Computation*, 51(184):699–706, 1988.

[60] G. E. Forsythe and W. R. Wasow. *Finite Difference Methods for PDEs*. John Wiley, New York, NY, 1960.

[61] R. Freund and N. Nachtigal. "QMR: A Quasi-Minimum Residual Method for Non-Hermitian Linear Systems". *Numerical Mathematics*, 60:315–339, 1991.

[62] R. Glowinski, G. H. Golub, G. A. Meurant, and J Périaux, editors. *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*. SIAM, Philadelphia, PA, 1988.

[63] Gene Golub and Charles F. Van Loan. *Matrix Computations*. John Hopkins University Press, Baltimore, MD, 1989.

[64] S. Gómez, J. P. Hennart, and R. A. Tapia, editors. *Advances in Numerical Partial Differential Equations and Optimization*, SIAM, Philadelphia, PA, 1989.

[65] William D. Gropp. *"Parallel Computing and Domain Decomposition"*. MCS P726, Argonne National Laboratory, 1991.

[66] William D. Gropp and David E. Keyes. *"Domain Decomposition on Parallel Computers"*. DCS RR 723, Yale University Research Report, 1989.

[67] William D. Gropp and David E. Keyes. *"Domain Decomposition with Local Mesh Refinement"*. DCS RR 726, Yale University Research Report, 1990.

[68] William D. Gropp and David E. Keyes. "Parallel Domain Decomposition and the Solution of Nonlinear Systems of Equations". In R. Glowinski, G. H. Golub, G. A. Meurant, and J Périaux, editors, *Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations*, SIAM, Philadelphia, PA, 1991.

[69] Wolfgang Hackbusch. *Theorie und Numerik elliptischer Differentialgleichungen*. Teubner Studienbücher : Mathematik, Stuttgart, Germany, 1986.

[70] L. A. Hageman and David M. Young. *Applied Iterative Methods*. Academic Press, New York, 1981.

[71] Matthias Heinkenschloss. *Krylov Subspace Methods for the Solution of Linear Systems and Linear Least Squares Problems*. Lecture Notes, 1994.

[72] Peter Henrici. *Elements of Numerical Analysis*. John Wiley and Sons, New York, NY, 1963.

[73] M. Hestenes and E. Stiefel. "Methods of Conjugate Gradients for Solving Linear Systems". *Jour. Res. National Bureau of Standards*, 49:409–436, 1952.

[74] David Hoff and Joel Smoller. "Error Bounds for Finite Difference Approximations for a Class of Nonlinear Parabolic Systems". *Mathematics of Computation*, 45(171):35–49, 1985.

[75] Alston S. Householder. *The Theory of Matrices in Numerical Analysis*. Blaisdell Publishing, New York, NY, 1965.

[76] E. N. Houstis and T. S. Papatheodorou. "High-Order Fast Elliptic Equation Solver". *ACM Trans. Mathematics Software*, 5(4):431–441, 1979.

[77] Wayne Joubert. "On the Convergence Behavior of the Restarted GMRES Algorithm for Solving Nonsymmetric Linear Systems". *Journal of Numerical Linear Algebra and Applications*, Submitted, 1992.

[78] Wayne Joubert, Thomas A. Manteuffel, Seymour Parter, and Sze–Ping Wong. "Preconditioning Second–Order Elliptic Operators: Experiment and Theory". *SIAM Journal of Scientific and Statistical Computing*, 13(1):259–288, 1992.

[79] L. V. Kantorovich and V. I. Krylov. *Approximate Methods of Higher Analysis*. Noordhoff Interscience, New York, NY, 1958.

[80] David Keyes. "Domain Decomposition: A Bridge Between Nature and Parallel Computers". In *Proceedings of the Symposium on Adaptive Multilevel and Hierarchical Computational Strategies*, ASME Winter Annual Meeting, Anaheim, CA, 1992.

[81] David E. Keyes and William D. Gropp. "Domain-Decomposable Preconditioners for Second-Order Upwind Discretizations of Multicomponent Systems". In R. Glowinski, G. H. Golub, G. A. Meurant, and J Périaux, editors, *Fourth International Symposium on Domain Decomposition Methods for Partial Differential Equations*, SIAM, Philadelphia, PA, 1991.

[82] Hung-Ju Kuo and Neil S. Trudinger. "Linear Elliptic Difference Inequalities with Random Coefficients". *Mathematics of Computation*, 55(191):37–53, 1990.

[83] Yu A. Kuznetsov. "Multi-Level Domain Decomposition Methods". *Applied Numerical Mathematics*, 6:303–314, 1989–1990.

[84] C. Lanczos. "An Iteration Method for the Solution of the Eigenvalue Problem of Linear Differential and Integral Operators". *Jour. Res. National Bureau of Standards*, 45:255–282, 1950.

[85] C. Lanczos. "Solution of Systems of Linear Equations by Minimized Iterations". *Jour. Res. National Bureau of Standards*, 49:33–53, 1952.

[86] L. Lapidus and W. E. Schiesser, editors. *Numerical Methods for Differential Systems*. Academic Press, New York, NY, 1976.

[87] Zi-Cai Li and Rudolf Mathon. "Error and Stability Analysis of Boundary Methods for Elliptic Problems with Interfaces". *Mathematics of Computation*, 54(189):41–61, 1990.

[88] Robert E. Lynch. "Fundamental Solutions of Nine Point Discrete Laplacians". *Applied Numerical Mathematics*, 10:325–334, 1992.

[89] Robert E. Lynch and John R. Rice. "A High-Order Difference Method for Differential Equations". *Mathematics of Computation*, 34(150):333–372, 1980.

[90] Robert E. Lynch and John R. Rice. "High Accuracy Finite Difference Approximations to Solutions of Elliptic Partial Differential Equations". *National Academic and Scientific, USA*, 75(6):2541–2544, 1978.

[91] Robert E. Lynch and John R. Rice. "The HODIE Method and its Performance for Solving Elliptic Partial Differential Equations". In C. de Boor and G. H. Golub, editors, *Recent Advances in Numerical Analysis*, pages 143–175, Academic Press, New York, NY, 1978.

[92] Robert E. Lynch and Donald H. Thomas. "Direct Solution of PDEs by Tensor Product Methods". *Numeriische Mathematik*, 6:185–199, 1964.

[93] Thomas A. Manteuffel. "The Tchebychev Iteration for Nonsymmetric Linear Systems". *Numerische Mathematik*, 28:307–327, 1977.

[94] Peter A. Markowich and Miloš A. Zlámal. "Inverse Average Type Finite Element Discretizations of Selfadjoint Second Order Elliptic Problems". *Mathematics of Computation*, 51(184):431–449, 1988.

[95] Gérard Meurant. "Domain Decomposition Methods for Partial Differential Equations on Parallel Computers". *International Journal of Super Computing Applications*, 2(4):5–12, 1988.

[96] A. R. Mitchell and D. F. Griffiths. *The Finite Difference Method in Partial Differnetial Equations*. John Wiley and Sons, New York, NY, 1980.

[97] B. Nour–Omid and B. N. Parlett. "Element Preconditioning using Splitting Techniques". *SIAM Journal of Scientific and Statistical Computing*, 6(3):761–770, 1985.

[98] J. Tinsley Oden. *Applied Function Analysis*. Prentice–Hall, Inc., Englewood Cliffs, NJ, 1979.

[99] John R. Rice. *Matrix Computations and Mathematical Software*. McGraw Hill, New York, NY, 1981.

[100] John R. Rice and Ronald F. Boisvert. *Solving Elliptic Problems Using ELLPACK*. Springer–Verlag, New York, NY, 1984.

[101] Garry Rodrigue and Donald Wolitzer. "Preconditioning by Complete Block Cyclic Reduction". *Mathematics of Computation*, 42(166):549–565, 1984.

[102] Y. Saad. "Krylov Subspace Methods for Solving Large Unsymmetric Linear Sustems". *Mathematics of Computation*, 37(155):105–126, 1981.

[103] Y. Saad and M. H. Schultz. "GMRES: A Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems". *SIAM Journal of Scientific and Statistical Computing*, 7:865–869, 1986.

[104] J. M. Sanz-Serna and C. Palencia. "A General Equivalence Theorem in the Theory of Discretization Methods". *Mathematics of Computation*, 45(171):143–152, 1985.

[105] W. E. Schlessor. *The Numerical Method of Lines*. Academic Press, New York, NY, 1991.

[106] John N. Shadid and Ray S. Tuminaro. "A Comparison of Preconditioned Nonsymmetric Krylov Methods on a Large–Scale MIMD Machine". *Sandia National Laboratory*, SAND91(0333), 1991.

[107] Barry F. Smith and Olof B. Widlund. "A Domain Decomposition Algorithm Using a Heirarchical Basis". *SIAM Journal of Scientific and Statistical Computing*, 11(6):1212–1220, 1990.

[108] P. Sonneveld. " CGS: A Fast Lanczos-Type Solver for Non-Symmetric Linear Systems". *SIAM Journal Sci. Statist. Comput.*, 10:36–52, 1989.

[109] R. Southwell. *Relaxation Methods in Theoretical Physics*. Clarendon Press, Oxford, 1946.

[110] Endre S̈uli, Boš Jovanović, and Lav Ivanović. "Finite Difference Approximations of Generalized Solutions". *Mathematics of Computation*, 45(172):319–327, 1985.

[111] Manil Suri. "On the Stability and Convergence of Higher–Order Mixed Finite Element Methods for Second–Order Elliptic Problems". *Mathematics of Computation*, 54(189):1–19, 1990.

[112] Kathryn Turner and Homer F. Walker. "Efficient High Accuracy Solutions with GMRES". *SIAM Journal of Scientific and Statistical Computing*, 13(3):815–825, 1992.

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

[114] V. Vemuri and Walter J. Karplus. *Digital Computer Treatment of Partial Differential Equations*. Prentice–Hall, Englewood Cliffs, NJ, 1981.

[115] Robert Vichnevetsky. *Computer Methods for Partial Differential Equations Volume 1*. Prentice–Hall, Englewood Cliffs, NJ, 1981.

[116] E. A. Volkov. *Chislennye Methody (Numerical Methods)*. Nauka, Moscow, 1986.

[117] H. Van Der Vorst. " BiCGSTAB:A Fast and Smoothly Converging Variant of BiCG for the Solution of Non-Symmetric Linear Systems". *SIAM Journal Sci. Statist. Comput.*, 13:631–644, 1992.

[118] O. B. Widlund. "Iterative Substructuring Methods: Algorithms and Theory for Elliptic Problems in the Plane". In R. Glowinski, G. H. Golub, G. A. Meurant, and J Périaux, editors, *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, SIAM, Philadelphia, PA, 1988.

[119] Stephen Wolfram. *Mathematica*. Addison-Wesley, New York, NY, 1988.

[120] Jinchao Xu and Xiao–Chuan Cai. "A Preconditioned GMRES Method for Nonsymmetric or Indefinite Problems". *Mathematics of Computation*, 59(200):311–319, 1992.

[121] David Young. "Iterative Methods for Solving PDEs of Elliptic Type". *Transactions of the American Math. Society*, 76:92–111, 1954.

[122] David Young. *Iterative Solution of Large Linear Systems.* Academic Press, New York, NY, 1971.

[123] David M. Young and Robert Todd Gregory. *A Survey of Numerical Mathematics.* Addison-Wesley, Readinf, MS, 19??

[124] E. C. Zachmanoglou and Dale W. Thoe. *Introduction to Partial Differential Equations with Applications.* Dover Publications, New York, NY, 1986.

VITA

George Gustav Pitts was born in Fayetteville, NC on February 18, 1944. He lived in Europe until he was sixteen. In 1966 he received an B.S. degree in Mathematics from the Virginia Polytechnic Institute and State University. In 1968 he received an M.S. degree in Mathematics from the Virginia Polytechnic Institute and State University. After over twenty years in industry, Mr. Pitts returned to graduate school and received his Ph.D. in Mathematics in 1994 from the Virginia Polytechnic Institute and State University.

Pitts is a member of the Association for Computing Machinery (ACM), the Society for Industrial and Applied Mathematics (SIAM), the American Mathematical Society (AMS), and the national honory mathematics society Pi Mu Epsilon. He is interested in continuing his research in numerical analysis.