SIAM J. SCI. COMPUT.
Vol. 19, No. 1, pp. 206–226, January 1998

015

AN EFFICIENT ITERATIVE METHOD FOR THE GENERALIZED STOKES PROBLEM*

VIVEK SARIN† AND AHMED SAMEH†

Abstract. The generalized Stokes problem, which arises frequently in the simulation of time-dependent Navier–Stokes equations for incompressible fluid flow, gives rise to symmetric linear systems of equations. These systems are indefinite due to a set of linear constraints on the velocity, causing difficulty for most preconditioners and iterative methods. This paper presents a novel method to obtain a preconditioned linear system from the original one which is then solved by an iterative method. This new method generates a basis for the velocity space and solves a reduced system which is symmetric and positive definite. Numerical experiments indicating superior convergence compared to existing methods are presented. A natural extension of this method to elliptic problems is also proposed, along with theoretical bounds on the rate of convergence, and results of experiments demonstrating robust and effective preconditioning.

Key words. Stokes problem, saddle point, iterative methods, preconditioning, mixed finite elements

AMS subject classifications. 65F10, 65N22, 65N30

PII. S106482759630365X

1. Introduction.

Large-scale simulation of incompressible fluid flow requires the solution of the nonlinear time-dependent Navier–Stokes equations. The most time-consuming part of this process is the solution of the generalized Stokes problem at each nonlinear iteration. Therefore, efficient algorithms for the generalized Stokes problem are indispensable for the numerical solution of the Navier–Stokes equations.

Given a domain Ω with boundary $\partial\Omega$ along with a function $\mathbf{f}$, the generalized Stokes problem requires finding the velocity $\mathbf{u}$ and pressure p satisfying

$$\alpha\mathbf{u} - \nu\boldsymbol{\Delta}\mathbf{u} + \nabla p = \mathbf{f} \text{ in } \Omega, \tag{1.1}$$

$$\nabla \cdot \mathbf{u} = 0 \text{ in } \Omega, \tag{1.2}$$

where α and ν are positive parameters. For simplicity, let

$$\mathbf{u} = 0 \text{ on } \partial\Omega. \tag{1.3}$$

The most commonly used Galerkin-type weak formulation for the generalized Stokes problem (1.1)–(1.3) is the following: given $\mathbf{f}$, we seek $\mathbf{u} \in \mathbf{H}_0^1(\Omega)$ and $p \in L_0^2(\Omega)$ such that

$$\begin{aligned} \alpha(\mathbf{u},\mathbf{v}) + \nu a(\mathbf{u},\mathbf{v}) + b(\mathbf{v},p) &= (\mathbf{f},\mathbf{v}) \quad \forall \mathbf{v} \in \mathbf{H}_0^1(\Omega), \\ b(\mathbf{u},q) &= 0 \quad \forall q \in L_0^2(\Omega), \end{aligned}$$

where

$$a(\mathbf{u},\mathbf{v}) = \int_\Omega \nabla\mathbf{u} : \nabla\mathbf{v} d\Omega \quad \forall \mathbf{u},\mathbf{v} \in \mathbf{H}^1(\Omega),$$

$$b(\mathbf{v},q) = -\int_\Omega q\nabla\cdot\mathbf{v} d\Omega \quad \forall \mathbf{v} \in \mathbf{H}^1(\Omega).$$

*Received by the editors May 15, 1996; accepted for publication (in revised form) March 22, 1997. This research was supported in part by NSF grants NSF/ECS 9527123 and NSF/CDA 9396332-001 and ARPA grant DOC/60NANB4D1615.

http://www.siam.org/journals/sisc/19-1/30365.html

†Department of Computer Science, Purdue University, West Lafayette, IN 47907-1398 (sarin@cs.purdue.edu, sameh@cs.purdue.edu).

Mixed finite element schemes [12, 7] used for the generalized Stokes problem require finding $\mathbf{u} \in \mathbf{V}_0^h$ and $p \in S_0^h$ for suitable spaces $\mathbf{V}_0^h$ and S_0^h such that

$$\begin{aligned} \alpha(\mathbf{u}, \mathbf{v}) + \nu a(\mathbf{u}, \mathbf{v}) + b(\mathbf{v}, p) &= (\mathbf{f}, \mathbf{v}) \quad \forall \mathbf{v} \in \mathbf{V}_0^h, \\ b(\mathbf{u}, q) &= 0 \quad \forall q \in S_0^h. \end{aligned}$$

In matrix notation, we need to solve the following system:

$$\begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}, \tag{1.4}$$

where $A = \alpha M + \nu T$, in which M is the $n \times n$ mass matrix and T is a symmetric positive definite matrix corresponding to the Laplace operator. B is an $n \times k$ matrix ($k < n$) such that B^T ensures the constraint of discrete null divergence on the solution vector u. In the case of *steady-state* flow, $\alpha = 0$ and $A = \nu T$.

The linear system (1.4) is a saddle-point problem. Even though the matrix A is symmetric and positive definite, the system is indefinite due to the incompressibility constraint. This causes difficulties both for iterative methods and commonly used preconditioners.

Most iterative methods proposed for (1.4) are modifications of Uzawa's method. The classical Uzawa method [11] solves the system

$$B^T A^{-1} B p = B^T A^{-1} f$$

by the method of steepest descent. Each iteration requires the solution of the linear system $Ax = b$. If an iterative method is used to solve $Ax = b$, then we obtain a two-level solver with inner and outer iterations. Convergence may be considerably improved by replacing A by $A_\epsilon = A + \frac{1}{\epsilon} B B^T$, where ϵ is a very small constant. This, however, worsens the condition number of A_ϵ and causes extreme difficulties for the inner iterative method.

The conjugate gradient method (CG) may be used to accelerate the convergence of the outer iterations. For a mixed finite element that satisfies the *inf-sup* condition (see [7]), the condition number of $B^T A^{-1} B$ is independent of the mesh discretization, and the CG algorithm converges rapidly. The inner system may be solved to desired accuracy in many ways including multigrid, spectral methods, and preconditioned CG algorithm. Verfürth [19] uses multigrid to solve the inner system $Ax = b$. Elman and Golub [10] analyze inexact solve at the inner level. Silvester and Wathen [20, 18] suggest diagonal and block preconditioners for solving the system (1.4).

In contrast, for the generalized Stokes problem, the condition number of $B^T A^{-1} B$ depends on the mesh discretization. In this case, preconditioners proposed by Cahouet and Chabard [8] and analyzed by Bramble and Pasciak [5] may be used. Other variants of the Uzawa method for such systems have been proposed in [4, 9, 16].

Projection methods [17, 6] obtain the velocity by solving

$$PAPu = Pf,$$

where $P = I - B(B^T B)^{-1} B^T$ is the orthogonal projection onto $Null(B^T)$. Computing the action of the projector in each iteration requires solving the system $B^T B x = b$ which is as difficult as solving the system $Au = b$, and may require the use of an inner iterative method. In the absence of effective preconditioners, these methods are competitive only for systems where A is diagonally dominant.

For realistic problems on two- and three-dimensional domains, both Uzawa-type and projection methods are two-level nested iterative methods with expensive iterations. Moreover, effective preconditioners for these methods have been developed only for the cases when the system satisfies certain restrictive conditions.

In this paper, we describe a method to compute a well-conditioned basis for the velocity space and solve a reduced symmetric positive definite linear system using the CG algorithm. The construction of the basis as well as the matrix–vector product in each iteration of CG require only $O(n)$ operations. Furthermore, our choice of the basis preconditions the reduced system implicitly, thereby accelerating the convergence of CG. Our method yields a single-level (nonnested) preconditioned iterative method with superior convergence compared to some of the existing methods. To the best of our knowledge, other techniques for computing divergence-free bases (e.g., [13, 14]) have not addressed similar issues.

Mixed finite element approximations of self-adjoint elliptic partial differential equations yield systems of the form (1.4) with $A = M$. This observation leads to a straightforward extension of our method to these problems and provides a strategy for robust and effective preconditioning for such systems. Moreover, our method can easily be extended to the commonly used finite difference and finite element approximations of these problems. This paper also presents application of our method to such problems, along with analysis of the preconditioned system and numerical experiments which confirm excellent preconditioning properties.

This paper is organized as follows. Sections 2 and 3 describe a multilevel scheme to compute the basis for $Null(B^T)$, along with extensions to elliptic problems and analysis of the condition number for these systems. Numerical experiments are presented in section 4, followed by conclusions and prospects for future research in section 5.

2. Multilevel scheme. Although the multilevel scheme proposed in this section is motivated by the generalized Stokes problem (1.4), it can be used to obtain preconditioned systems for self-adjoint elliptic partial differential equations. It may be noted, however, that our approach has no obvious similarities with other well-known multilevel methods for elliptic problems (see [1, 21, 2, 3]).

A multilevel approach is used to compute a basis for $Null(B^T)$. This null-space basis P_2 is expressed as the product of a sequence of sparse matrices, such that it requires only $O(n)$ operations to perform a matrix–vector product with P_2. In fact, we determine matrices P, D, and Z such that

$$P^T B Z = \begin{pmatrix} D \\ 0 \end{pmatrix},$$

where P is a nonsingular $n \times n$ matrix, D is a $k \times k$ diagonal matrix, and Z is a $k \times k$ orthogonal matrix. Further, $P = [P_1, P_2]$, where P_2 is comprised of the last $n - k$ columns of P. The linear system (1.4) may be rewritten as

$$\begin{pmatrix} P_1^T A P_1 & P_1^T A P_2 & D \\ P_2^T A P_1 & P_2^T A P_2 & 0 \\ D & 0 & 0 \end{pmatrix} \begin{pmatrix} \hat{u}_1 \\ \hat{u}_2 \\ Z^T p \end{pmatrix} = \begin{pmatrix} P_1^T f \\ P_2^T f \\ 0 \end{pmatrix},$$

where $u = P\hat{u}$. The solution is obtained by solving the reduced system

$$P_2^T A P_2 \hat{u}_2 = P_2^T f \tag{2.1}$$

by an iterative method like the CG algorithm.

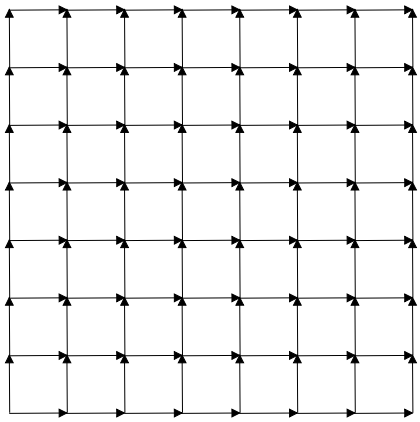

FIG. 2.1. *A uniform mesh with directional edges for the discrete gradient operator E.*

2.1. Sample problem: The Poisson equation. As an illustration, we describe our scheme for a finite difference approximation of the following Poisson equation on a two-dimensional square domain:

$$-\nabla \cdot \nabla u = f \text{ in } \Omega, \tag{2.2}$$

$$\frac{\partial u}{\partial n} = 0 \text{ on } \partial\Omega \tag{2.3}$$

which has a unique solution under the assumption that f has zero mean. It may be noted that the proposed multilevel scheme is more general and can easily be extended to more complicated PDEs with different boundary conditions, as well as other discretization schemes like the finite element and finite volume methods.

Equation (2.2) may written in the following form:

$$\begin{aligned} \mathbf{v} + \nabla u &= 0, \\ -\nabla \cdot \mathbf{v} &= -f, \end{aligned}$$

where $\mathbf{v}$ is the gradient of the function u. The domain is discretized by a $\sqrt{n} \times \sqrt{n}$ uniform mesh (Fig. 2.1), where $n = 4^k$ for some constant k is assumed for simplicity. Let us assume that $\mathbf{v} = (v_1, v_2)^T$, where v_1 and v_2 are the components of the gradient along the axes and are fixed at the midpoint of edges along the x- and y-axis, respectively. Furthermore, the boundary condition on u implies that $\mathbf{v} = 0$ outside the domain. Using one-sided differencing, the equation at the node (i, j) is

$$\begin{aligned} v_{i,j,1} + \frac{u_{i+1,j} - u_{i,j}}{h} &= 0, \\ v_{i,j,2} + \frac{u_{i,j+1} - u_{i,j}}{h} &= 0, \\ -\left(\frac{v_{i,j,1} - v_{i-1,j,1}}{h} + \frac{v_{i,j,2} - v_{i,j-1,2}}{h} \right) &= -f_{i,j}. \end{aligned}$$

Appropriate scaling of the variables yields the linear system

$$\begin{pmatrix} I & E \\ E^T & 0 \end{pmatrix} \begin{pmatrix} y \\ x \end{pmatrix} = \begin{pmatrix} 0 \\ -b \end{pmatrix}, \tag{2.4}$$

where E and E^T are discrete forms of the gradient operator (∇) and the divergence operator ($-\nabla\cdot$), respectively.

Further, E is a matrix of size $e \times n$, where n is the number of nodes and e is the number of edges in the mesh (in this case $e = 2(n - \sqrt{n})$). It may be noted that the rows of E correspond to edges in the mesh, and columns correspond to nodes. Natural ordering of nodes and natural ordering of edges along the x-direction followed by edges along the y-direction give

$$E = \begin{pmatrix} C & & & & & \\ & C & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & & & & C & \\ -I & I & & & & \\ & -I & I & & & \\ & & \ddots & \ddots & & \\ & & & -I & I & \end{pmatrix},$$

where I is a $\sqrt{n} \times \sqrt{n}$ identity matrix and C is the following $\sqrt{n} - 1 \times \sqrt{n}$ bidiagonal matrix:

$$C = \begin{pmatrix} -1 & 1 & & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{pmatrix}.$$

As mentioned earlier, the matrices P, D, and Z are to be determined such that

$$P^T E Z = \begin{pmatrix} D \\ 0 \end{pmatrix}, \tag{2.5}$$

where P is a nonsingular $e \times e$ matrix, D is an $n \times n$ diagonal matrix, and Z is an $n \times n$ orthogonal matrix. Assuming $P = [P_1, P_2]$, where P_2 consists of the last $e - n$ columns, the linear system (2.4) is transformed to

$$\begin{pmatrix} P_1^T P_1 & P_1^T P_2 & D \\ P_2^T P_1 & P_2^T P_2 & 0 \\ D & 0 & 0 \end{pmatrix} \begin{pmatrix} \hat{y_1} \\ \hat{y_2} \\ Z^T x \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ -Z^T b \end{pmatrix},$$

where $y = P\hat{y}$ and the solution x is given as

$$x = ZD^{-1}P_1^T[I - P_2(P_2^T P_2)^{-1}P_2^T]P_1 D^{-1} Z^T b.$$

Since the main computational effort involves the solution of a system of the type $P_2^T P_2 w = d$ by CG, it is necessary that $P_2^T P_2$ be well conditioned, and the matrix–vector product with $P_2^T P_2$ be computed in $O(n)$ operations. In the next section, we describe how to compute the matrices P, D, and Z with these desirable properties.

2.2. Multilevel scheme for Poisson equation. The multilevel scheme systematically coarsens the mesh to obtain the matrices P, D, and Z with the desired properties. A coarse level mesh is obtained from the original mesh by combining

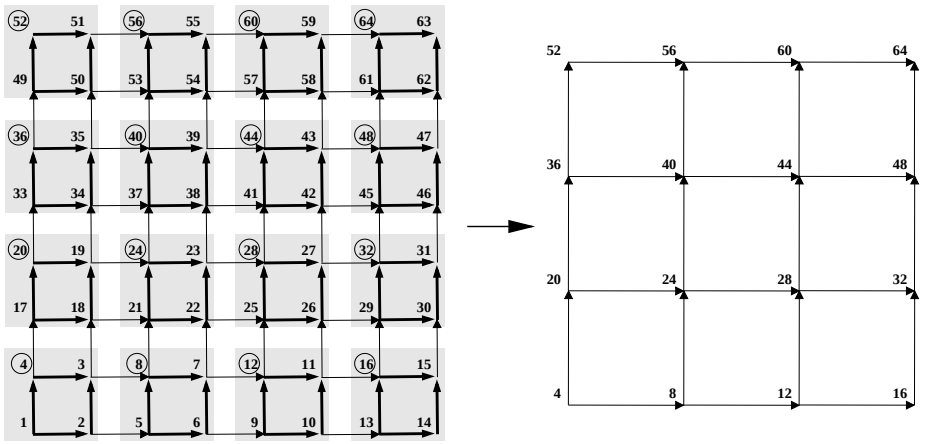

FIG. 2.2. *Groups of four nodes (shaded) are combined into a single node (circled) for the coarse level mesh.*

groups of four nodes (see Fig. 2.2) into a single node. The coarsening of the mesh is used to obtain the discrete gradient matrix $E^{(1)}$ for the coarse mesh from the one on the original mesh. The linear transformation from E to $E^{(1)}$ may be written as

$$P^{(1)^T} E Z^{(1)} = \begin{pmatrix} D^{(1)} \\ E^{(1)} \\ 0 \end{pmatrix},$$

where $D^{(1)}$ is a submatrix of D. The mesh is recursively coarsened to a single node by repeated linear transformations to obtain the discrete gradient matrix $E^{(i+1)}$ for level $i+1$ from $E^{(i)}$ at the previous level. The overall transformation is given by

$$P = P^{(1)} P^{(2)} \cdots P^{(k)},$$
$$Z = Z^{(1)} Z^{(2)} \cdots Z^{(k)},$$

where $k = \log_4 n$.

We now describe the transformations to obtain $E^{(1)}$ from E. The nodes in the mesh are grouped into $\frac{n}{4}$ partitions, as shown in Fig. 2.2, and reordered so that nodes within a partition are ordered consecutively. The edges are divided into two classes: *interior* edges connect nodes within the same partition whereas *boundary* edges connect nodes in different partitions. Figure 2.2 shows that each partition has four interior edges, and each pair of neighboring partitions has two boundary edges between them.

Further, the edges (rows of E) are reordered such that the interior edges within a partition are ordered consecutively, followed by all the boundary edges. After reordering we have

$$E = \begin{pmatrix} E_{int} \\ E_{bnd} \end{pmatrix},$$

where E_{int} consists of the rows of E corresponding to the interior edges, and E_{bnd} consists of the rows corresponding to the boundary edges. E_{int} is the following block

diagonal matrix:

$$E_{int} = \begin{pmatrix} E_{int_1} & & \\ & \ddots & \\ & & E_{int_r} \end{pmatrix},$$

where r is the number of partitions, and

$$E_{int_i} = \begin{pmatrix} -1 & 1 & & \\ & -1 & 1 & \\ & & 1 & -1 \\ -1 & & & 1 \end{pmatrix}, \quad i = 1, \ldots, r.$$

The singular value decomposition of $E_{int_i},\ i = 1, \ldots, r$, is

$$E_{int_i} = U_i S_i V_i^T, \tag{2.6}$$

$$U_i = \begin{pmatrix} -\frac{1}{2} & \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & -\frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} \end{pmatrix}, \tag{2.7}$$

$$S_i = \begin{pmatrix} 2 & & & \\ & \sqrt{2} & & \\ & & \sqrt{2} & \\ & & & 0 \end{pmatrix}, \tag{2.8}$$

$$V_i^T = \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & 0 & -\frac{1}{\sqrt{2}} \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix}, \tag{2.9}$$

and that of E_{int} is

$$\begin{aligned} E_{int} &= USV^T \\ &= \begin{pmatrix} U_1 & & \\ & \ddots & \\ & & U_r \end{pmatrix} \begin{pmatrix} S_1 & & \\ & \ddots & \\ & & S_r \end{pmatrix} \begin{pmatrix} V_1^T & & \\ & \ddots & \\ & & V_r^T \end{pmatrix}. \end{aligned} \tag{2.10}$$

The coarsening process divides the nodes in the original mesh into two classes: *active* nodes which form the coarse mesh and *inactive* nodes. The matrices $P^{(1)}$ and $Z^{(1)}$ are used to obtain the discrete gradient operator $E^{(1)}$ for a coarse mesh defined only on the active nodes.

Let $S^\dagger$ be the pseudoinverse of S. Then

$$\begin{aligned} \begin{pmatrix} I & 0 \\ -E_{bnd} V S^\dagger & I \end{pmatrix} \begin{pmatrix} U^T & \\ & I \end{pmatrix} \begin{pmatrix} E_{int} \\ E_{bnd} \end{pmatrix} V &= \begin{pmatrix} S \\ E_{bnd} V (I - S^\dagger S) \end{pmatrix} \\ &= \begin{pmatrix} S \\ E_{bnd} \hat{V} \end{pmatrix}, \end{aligned} \tag{2.11}$$

where $\hat{V} = V(I - S^{\dagger}S)$ is the matrix V with nonzeros only in the columns corresponding to the $\frac{n}{4}$ active nodes. In particular, $\hat{V}$ consists of the last column of V_i for $i = 1, \ldots, r$.

The next step is to establish that rows of $E_{bnd}\hat{V}$ actually define edges in the coarse mesh. To show this, let us consider the nonzero structure of the matrix $E_{bnd}\hat{V}$. Clearly, $E_{bnd}\hat{V}$ has nonzeros only in the columns of the active nodes. Further, a boundary edge between two nodes in the original mesh E_{bnd} is transformed into an edge between the active nodes of those partitions in $E_{bnd}\hat{V}$, and scaled by $\frac{1}{2}$. For instance, in Fig. 2.2 the edge from 2 to 5 in E_{bnd} with value $(-1, 1)$ is replaced by an edge from 4 to 8 in $E_{bnd}\hat{V}$ with value $(-\frac{1}{2}, \frac{1}{2})$. This implies that each row of $E_{bnd}\hat{V}$ is an edge between the active nodes of a pair of neighboring partitions. Therefore, $E_{bnd}\hat{V}$ is the edge matrix for a coarse mesh defined by the active nodes.

The edge matrix $E_{bnd}\hat{V}$ for the coarse mesh can be simplified further, as the following observation suggests. Observe that there are exactly two edges between each pair of neighboring active nodes. For example, in Fig. 2.2 both the edges from 2 to 5 and 3 to 8 are replaced by edges from 4 to 8, each with value $(-\frac{1}{2}, \frac{1}{2})$. It can be shown that these rows of $E_{bnd}\hat{V}$ are linearly dependent[1]. The orthogonal transformation

$$Q_{i,j}^T = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$$

combines two linearly dependent identical rows of $E_{bnd}\hat{V}$ which correspond to the boundary edges between the active nodes of partitions i and j. As a result, one of the rows becomes zero. In the example of the two edges between 4 and 8,

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} -\frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix} = \begin{pmatrix} -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ 0 & 0 \end{pmatrix}.$$

Using $Q_{i,j}^T$, we define the block diagonal orthogonal matrix Q^T which operates on the rows of $E_{bnd}\hat{V}$ to combine linearly dependent identical rows between pairs of active nodes. Further, the rows of Q^T are permuted so that all the rows which are nonzero after multiplication with Q^T are collected at the top. Then we have

$$Q^T E_{bnd}\hat{V} = \begin{pmatrix} E^{(1)} \\ 0 \end{pmatrix}, \tag{2.12}$$

where $E^{(1)}$ is a gradient operator for the coarse mesh defined on the $\frac{n}{4}$ active nodes. In particular, this new gradient operator has been scaled by $\frac{1}{\sqrt{2}}$.

The linear transformation to obtain $E^{(1)}$ from E can be formally defined as follows:

$$\begin{aligned} Z^{(1)} &= V, \\ P^{(1)^T} &= \begin{pmatrix} I & \\ & Q^T \end{pmatrix} \begin{pmatrix} I & \\ -E_{bnd}VS^{\dagger} & I \end{pmatrix} \begin{pmatrix} U^T & \\ & I \end{pmatrix}. \end{aligned} \tag{2.13}$$

[1] A constant vector over two neighboring partitions i and j is in the null space of the submatrix of E formed by E_{int_i}, E_{int_j}, and $E_{bnd_{(i,j)}}$, where $E_{bnd_{(i,j)}}$ is the set of rows corresponding to the boundary edges between partitions i and j.

From (2.11) and (2.12) it follows that

$$P^{(1)^T} E Z^{(1)} = \begin{pmatrix} S \\ E^{(1)} \\ 0 \end{pmatrix}.$$

Recall that S is a diagonal matrix with zero elements on the diagonal for the columns corresponding to the active nodes. On the other hand, $E^{(1)}$ has nonzero columns only for the active nodes. This implies that we can choose suitable $P^{(2)}$ and $Z^{(2)}$ which operate only on the nonzero rows and columns of $E^{(1)}$ to obtain $E^{(2)}$ without affecting S. Using this technique, a coarse level gradient operator $E^{(i+1)}$ is obtained from $E^{(i)}$ at the ith level by the transformation

$$P^{(i+1)^T} E^{(i)} Z^{(i+1)} = \begin{pmatrix} S^{(i+1)} \\ E^{(i+1)} \\ 0 \end{pmatrix}.$$

This process may be continued until we reach a mesh with a single partition. At this level, say k, all the edges are interior edges and

$$P^{(k)} = \begin{pmatrix} U^{(k)} & \\ & I \end{pmatrix},$$

where $U^{(k)} S^{(k)} V^{(k)^T}$ is the SVD of the coarsest level gradient matrix $E^{(k)}$.

The combined transformation for all the k levels is

$$P = P^{(1)} P^{(2)} \cdots P^{(k)},$$
$$Z = Z^{(1)} Z^{(2)} \cdots Z^{(k)}, \text{ and}$$
$$P^T E Z = \begin{pmatrix} S^{(1)} \\ \vdots \\ S^{(k)} \\ 0 \end{pmatrix}.$$

The diagonal matrix $S^{(i)}$, obtained at the ith level, has nonzero elements which correspond exactly to those nodes which become inactive at this level. This also implies that each column of $P^T E Z$ has exactly one nonzero[2]. A suitable permutation of the rows of this matrix is used to convert it into the desired form (2.5).

To summarize, we have described a scheme which transforms the gradient matrix E for the original mesh into one for a coarser mesh. We have also shown how to apply this strategy recursively to coarsen the mesh all the way up to a single node. This yields a sequence of linear transformations which converts the original gradient matrix E to a diagonal matrix. An outline of the multilevel algorithm for computing the decomposition (2.5) is presented in the appendix.

The following lemma establishes that the computational complexity of matrix–vector products with Z and P is $O(n)$ operations.

LEMMA 2.1. *A matrix–vector product with Z and P is $O(n)$ operations.*

[2]For our sample problem with rank deficiency, the multilevel scheme terminates with zero as the last column of $S^{(k)}$. Since the last column of Z is the constant vector over the domain which is identical to $Null(E)$, a solution can still be found provided b is orthogonal to $Null(E)$, i.e., if b has zero mean.

Proof. The proof is in two parts. First we show that a matrix–vector product with the first level matrices $Z^{(1)}$ and $P^{(1)}$ requires $O(n)$ operations, and then we prove that the complexity of a matrix–vector product with Z or P is $O(n)$ operations. From the preceding discussion, it is clear that U and V are $n \times n$ block diagonal matrices with 4×4 dense blocks. E_{bnd} is a $q \times n$ matrix with two nonzeros per row, where q is the number of boundary edges ($q \approx n$). Furthermore, Q^T is a $q \times q$ matrix with two nonzeros in each row. Therefore, matrix–vector products with $Z^{(1)}$ and $P^{(1)}$ each require $O(n)$ operations. Since the number of nodes in the mesh decreases by a factor of four at each level of coarsening, the total number of operations for a matrix–vector product with Z or P is $\sum_{i=0}^{\log_4 n-1} \frac{cn}{4^i}$ for some constant c, which is $O(n)$. ∎

We conclude this section with remarks on the versatility of the proposed scheme.

Remark 1. A truncated multilevel scheme. In section 3, we show that for our sample problem the condition number $\kappa(P_2^T P_2)$ degrades by a constant factor with each level. Substantial improvement in convergence may be achieved by terminating the multilevel scheme after $k_0 < k$ levels. This *truncated* multilevel scheme requires the QR factorization of the gradient matrix $E^{(k_0)}$ at the coarsest level. Now, $P^{(k_0)} = Q$, and D has the upper triangular matrix R in the columns for the active nodes at level k_0. If we choose $k_0 = \frac{k}{2}$, $\kappa(P_2^T P_2)$ is reduced by a power of $\frac{1}{2}$, along with an increase in storage and computation per iteration of only $O(n)$. However, experimental evidence in section 4 suggests that the condition number for the truncated multilevel scheme for $k_0 = \frac{1}{2}\log_4 n$ is $O(\sqrt[4]{n})$.

Remark 2. Dirichlet boundary conditions. Dirichlet boundary condition at a node i results in the removal of that column from the gradient matrix E. Due to this, each neighbor j of i acquires a *self edge* which corresponds to a row of E with a single nonzero element in the column for j. Even though a partition including such a neighbor j has a nonzero smallest singular value, it is not used to define $S^\dagger$. Instead, it is included in the gradient matrix for the coarse mesh $E^{(1)}$ and acts as a Dirichlet boundary condition at the active node from this partition.

Combining the edges between two partitions by orthogonal transformation Q^T can also yield self edges for the active nodes on the boundary. This is due to the fact that in $E_{bnd}\hat{V}$, the boundary edges between such nodes are no longer linearly dependent. For example, if active nodes i (from a boundary partition) and j are connected by a set of boundary edges which form the submatrix $[E_{bnd}\hat{V}]_{i,j}$ with linearly independent rows, then $Q_{i,j}^T$ is chosen such that $Q_{i,j}^T[E_{bnd}\hat{V}]_{i,j} = R_{i,j}$, where $R_{i,j}$ is an upper triangular matrix. The first row of $R_{i,j}$ is the new edge between the nodes i and j. The second row (with a single nonzero element) is a new self edge for node j and acts as a Dirichlet condition at this node. Recall that in the case of Neumann conditions the rows of $[E_{bnd}\hat{V}]_{i,j}$ were linearly dependent, and the second row of $R_{i,j}$ was zero.

The number of operations required for matrix–vector products with P and Z are still $O(n)$. The estimate for the condition number $\kappa(P_2^T P_2)$ derived in section 3 is valid for Dirichlet boundary conditions as well.

Remark 3. Multilevel scheme for finite elements. The finite element method yields a linear system with the matrix

$$A = \sum_{j=1}^{t} A_j,$$

where A_j is the element stiffness matrix for the jth element, and t is the total number of elements. Further, A_j is rank deficient and may be expressed as $A_j = V_j V_j^T$. Defining $E^T = [V_1, V_2, \ldots, V_t]$, we can express the linear system in the alternate

form (2.4). The main difference now is that each row of E consists of three nonzeros corresponding to the nodes of each triangular element. However, these rows can be viewed as *super edges* of size 3, where a super edge of size m is defined as an edge over m nodes. As before, the nodes are grouped into partitions. The elements within a partition are called *interior elements*, and the elements lying outside the partitions are called *boundary elements*. Since each element had two super edges (rows of V_j^T), we can partition the rows of E into interior rows E_{int} and boundary rows E_{bnd}. The linear transformations to obtain $E^{(1)}$ from E are derived in a manner similar to the one described previously.

From this point onward, the multilevel scheme for the finite element method is essentially the same as that for the finite difference scheme. The mesh is systematically coarsened, and the matrices $E^{(i)}$ are obtained in the same manner. The rows of $E^{(i)}$ may have two nonzeros for edges between two nodes, or three nonzeros for super edges arising from elements.

3. Convergence. In this section, we prove that the condition number of the matrix $P_2^T P_2$ for our sample problem is smaller than that of the original system, which confirms the fact that the multilevel scheme acts as an effective preconditioner as well. It is well known that the rate of convergence of CG for solving the symmetric positive definite system $Ax = b$ is given by

$$\frac{\|e_m\|}{\|e_0\|} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^m,$$

where $\|e_m\|_A = \|x - x_m\|_A$ is the A-norm of the error in the solution at the mth iteration, and κ is the condition number of A. Therefore, CG converges rapidly for systems with small κ.

An estimate of $\kappa(P_2^T P_2)$ is obtained as follows. First we represent P_2 as a product of matrices $P_2^{(i)}, i = 1, \ldots, k$, where $P_2^{(i)}$ is a submatrix of $P^{(i)}$ at the ith level of the multilevel scheme. Next, we estimate the extremal singular values of $P_2^{(i)}$, say $\sigma_{max}(P_2^{(i)})$ and $\sigma_{min}(P_2^{(i)})$. Then

$$\kappa(P_2^T P_2) \leq \prod_{i=1}^{k} \left(\frac{\sigma_{max}(P_2^{(i)})}{\sigma_{min}(P_2^{(i)})}\right)^2, \tag{3.1}$$

where P_2 consists of $e-n$ columns of P which form a basis for $Null(E^T)$, i.e., $P_2^T E = 0$. Since

$$P^{(1)^T} E Z^{(1)} = \begin{pmatrix} S \\ E^{(1)} \\ 0 \end{pmatrix},$$

the columns of $P^{(1)}$ are divided into three categories: *nonbasis* columns that yield a nonzero row in S, *basis* columns that yield a zero row, and *undetermined* columns that yield the nonzero matrix $E^{(1)}$. Subsequent transformations using $P^{(i)}, i = 2, \ldots, k$, applied to $E^{(1)}$ do not affect basis and nonbasis columns. This leads to the conclusion that the basis columns are included in P_2 and nonbasis columns are excluded from P_2. At this point, no such assertion can be made for the undetermined columns. At level i the equation

$$P^{(i)^T} E^{(i-1)} Z^{(i)} = \begin{pmatrix} S^{(i)} \\ E^{(i)} \\ 0 \end{pmatrix}$$

is used to divide columns of $P^{(i)}$ into basis, nonbasis, and undetermined columns, and the basis columns are included in P_2. The last level has only basis and nonbasis columns. The matrix P_2 is defined as follows: let $P_2^{(i)}$ be the set of basis and undetermined columns of $P^{(i)}$ at level i. Then

$$P_2 = P_2^{(1)} P_2^{(2)} \cdots P_2^{(k)}. \tag{3.2}$$

Assuming

$$W = -{S^\dagger}^T V^T E_{bnd}^T Q$$

we obtain from (2.13)

$${P^{(1)}}^T P^{(1)} = \begin{pmatrix} I & \\ W^T & Q^T \end{pmatrix} \begin{pmatrix} I & W \\ & Q \end{pmatrix} = \begin{pmatrix} I & W \\ W^T & I + W^T W \end{pmatrix}.$$

Notice that the off-diagonal blocks contain nonzero elements *only* in the nonbasis rows (and columns) for the first level. To obtain ${P_2^{(1)}}^T P_2^{(1)}$ we extract the rows and columns corresponding to basis and undetermined rows. This yields the following $(e - \frac{3n}{4}) \times (e - \frac{3n}{4})$ matrix:

$${P_2^{(1)}}^T P_2^{(1)} = \begin{pmatrix} I & \\ & I + W^T W \end{pmatrix}.$$

The following lemma establishes bounds for the extremal singular values of $P_2^{(i)}$.

LEMMA 3.1. *Define* $W^{(i)} = {S^{(i)}}^{\dagger^T} {V^{(i)}}^T {E_{bnd}^{(i-1)}}^T Q^{(i)}$. *Then*

$$1 \le \sigma({P_2^{(i)}})^2 \le 1 + \sigma_{max}({W^{(i)}})^2.$$

Proof. From the above discussion, it follows that for level i

$${P_2^{(i)}}^T P_2^{(i)} = \begin{pmatrix} I & \\ & I + {W^{(i)}}^T W^{(i)} \end{pmatrix}$$

which proves the lemma. ∎

THEOREM 3.2. *The condition number of* $P_2^T P_2$

$$\kappa(P_2^T P_2) \le \prod_{i=1}^{k} [1 + \sigma_{max}(W^{(i)})^2].$$

Proof. The proof follows from (3.1), (3.2), and Lemma 3.1. ∎

We now estimate $\kappa(P_2^T P_2)$ for our sample problem. From the definition of $W^{(i)}$,

$$\sigma_{max}(W^{(i)}) \le \sigma_{max}({S^{(i)}}^\dagger) \sigma_{max}(E_{bnd}^{(i)}).$$

At the first level (2.10) gives $\sigma_{max}(S^\dagger) = \frac{1}{\sqrt{2}}$. To estimate $\sigma_{max}(E_{bnd})$, we compute the SVD of E_{bnd} in a manner similar to the SVD for E_{int}. Groups of *boundary* edges (see Fig. 3.1) are used to define partitions of nodes and reorder the rows and columns of E_{bnd} to obtain the following block diagonal matrix:

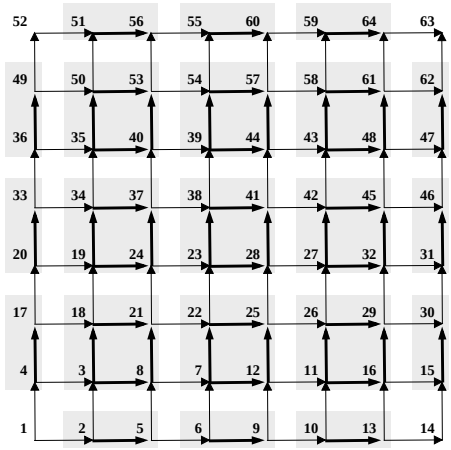

FIG. 3.1. *Groups of boundary edges (bold) which form partitions (shaded) in an* 8×8 *mesh.*

$$\tilde{E}_{bnd} = \begin{pmatrix} E_{bnd_1} & & & \\ & E_{bnd_2} & & \\ & & \ddots & \\ & & & E_{bnd_l} \end{pmatrix},$$

where l is the number of partitions. For partitions with two nodes

$$E_{bnd_j} = \begin{pmatrix} -1 & 1 \end{pmatrix}$$

and the SVD is

$$E_{bnd_j} = U_j S_j V_j^T = \begin{pmatrix} 1 \end{pmatrix} \begin{pmatrix} \sqrt{2} \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix},$$

whereas for partitions with four nodes

$$E_{bnd_j} = \begin{pmatrix} -1 & 1 & & \\ & -1 & 1 & \\ & & 1 & -1 \\ -1 & & & 1 \end{pmatrix}$$

and the SVD is

$$E_{bnd_j} = U_j S_j V_j^T,$$

where U_j, S_j, and V_j are identical to U_i, S_i, and V_i in (2.7)–(2.9). Therefore, the SVD of $\tilde{E}_{bnd}$ is given by

$$\begin{aligned} \tilde{E}_{bnd} &= U_{bnd} S_{bnd} V_{bnd}^T \\ &= \begin{pmatrix} U_1 & & \\ & \ddots & \\ & & U_l \end{pmatrix} \begin{pmatrix} S_1 & & \\ & \ddots & \\ & & S_l \end{pmatrix} \begin{pmatrix} V_1^T & & \\ & \ddots & \\ & & V_l^T \end{pmatrix} \end{aligned}$$

which shows that $\sigma_{max}(E_{bnd}) = 2$ and $\sigma_{max}(W) \leq \sqrt{2}$.

Recall that $E^{(1)}$ was the gradient matrix for the first level coarse mesh scaled by a factor $\frac{1}{\sqrt{2}}$. Each level of coarsening scales the gradient matrix by $\frac{1}{\sqrt{2}}$, which implies that at level i, $E^{(i)}$ is the gradient matrix for the coarse mesh scaled by a factor of $2^{-i/2}$. Hence, it may be concluded that for level i

$$\begin{aligned}
&\sigma_{max}({S^{(i)}}^{\dagger}) = 2^{(i-1)/2},\\
&\sigma_{max}(E_{bnd}^{(i)}) = 2^{(2-i)/2}, \text{ and}\\
&\sigma_{max}(W^{(i)}) \leq \sqrt{2}.
\end{aligned}$$

THEOREM 3.3. *For the Poisson equation with Neumann boundary conditions,*

$$\begin{aligned}
\sigma_{max}(W^{(i)}) &\leq \sqrt{2}, \quad i = 1, \ldots, \log_4 n,\\
\kappa(P_2^T P_2) &\leq n^{\frac{1}{2}\log_2 3}.
\end{aligned}$$

Proof. From the preceding discussion, $\sigma_{max}(W^{(i)}) \leq \sqrt{2}$ for $i = 1, \ldots, k$, where k is the total number of levels. Theorem 3.2 shows that

$$\kappa(P_2^T P_2) \leq \prod_{i=1}^{k} [1 + \sigma_{max}(W^{(i)})^2].$$

It follows that for $k = \log_4 n$

$$\kappa(P_2^T P_2) \leq 3^{\log_4 n} = n^{\frac{1}{2}\log_2 3} \approx n^{0.7925},$$

which proves the theorem. ∎

Remark 4. Theorem 3.3 gives a pessimistic bound for $\kappa(P_2^T P_2)$. Results of numerical experiments in section 4 suggest that $\kappa(P_2^T P_2)$ is $O(\sqrt{n})$. This is most likely due to the inaccuracy in estimating $\sigma_{max}(W^{(i)})$.

4. Numerical experiments. In this section, we present the results of numerical experiments for elliptic problems as well as the generalized Stokes problem. All the experiments were performed on an IBM RS6000 workstation with 66.5 MHz clock and 256 MB main memory.

4.1. Elliptic problems. The main purpose of this section is to illustrate the preconditioning properties of the proposed multilevel scheme and, therefore, we compare it with the diagonally preconditioned CG method (DCG). The following PDE was solved on a unit square domain with appropriate boundary conditions:

$$-\nabla \cdot (a(x, y)\nabla u) = f. \tag{4.1}$$

The domain is discretized by a uniform $\sqrt{n} \times \sqrt{n}$ mesh, and a finite difference approximation identical to the one described in section 2.1 is used. The convergence criterion for terminating an iteration was a 10^{-6} reduction in the relative residual norm for DCG. For the multilevel scheme, an iteration was terminated upon a 10^{-8} reduction in the relative residual norm, which yielded a relative residual norm less than 10^{-6} for the original system. In each case, the initial guess for the solution was zero.

Problem 1. Poisson equation. As our first example, we solve (4.1) with $a(x, y) = 1$ over the entire domain. Table 4.1 presents the comparison between the multilevel

TABLE 4.1
Comparison of DCG and multilevel scheme for the Poisson equation.

Mesh size	Neumann				Dirichlet			
	DCG		Multilevel		DCG		Multilevel	
($\sqrt{n}$)	Iter	Time	Iter	Time	Iter	Time	Iter	Time
16	77	.10	19	.10	39	.05	19	.12
32	157	.73	27	.53	77	.38	25	.48
64	329	6.37	38	3.19	153	3.09	35	2.83
128	666	51.76	53	17.50	309	24.59	49	16.18

TABLE 4.2
Comparison of DCG and multilevel scheme for different inhomogeneity functions.

Mesh size	Type a		Type b		Type c		Type d	
($\sqrt{n}$)	DCG	ML	DCG	ML	DCG	ML	DCG	ML
16	44	18	50	17	48	19	46	21
32	90	25	101	23	95	26	92	29
64	180	34	199	32	188	36	179	40
128	361	46	394	43	372	48	355	55

scheme and DCG method for the cases of Dirichlet and Neumann conditions over the entire boundary. The system was made nonsingular for the case with Neumann condition by enforcing a Dirichlet condition at node 1.

The number of iterations and time (in seconds) required for CG in DCG and multilevel scheme are reported in columns *Iter* and *Time*, respectively. It is well known that the linear system for the DCG method has condition number of $O(n)$. In contrast, the convergence of the multilevel scheme suggests that $\kappa(P_2^T P_2) = O(\sqrt{n})$. It was also observed that the time to compute the matrices P, Z, and D was a fifth of the overall time and increased linearly with the size of the problem, confirming that the computation required for the decomposition of the edge matrix E in the multilevel scheme is $O(n)$.

Problem 2. Inhomogeneous problems. We consider problems of the form (4.1) with Dirichlet boundary condition. We use the same discretization as in the previous case but allow the following types of functions:

1. Type a: $a(x,y) = \begin{cases} 0.01 & \text{if } x \le 0.5, \\ 1 & \text{if } x > 0.5. \end{cases}$
2. Type b: $a(x,y) = \begin{cases} 0.01 & \text{if } x \le 0.5 \text{ and } y < 0.5, \\ 1 & \text{if } x > 0.5 \text{ and } y < 0.5, \\ 0.0001 & \text{if } x \le 0.5 \text{ and } y \ge 0.5, \\ 0.01 & \text{if } x > 0.5 \text{ and } y \ge 0.5. \end{cases}$
3. Type c: $a(x,y) = 0.01 + x^2 + y^2$.
4. Type d: $a(x,y) = \begin{cases} 10^x & \text{if edge along } x\text{-axis}, \\ 10^y & \text{if edge along } y\text{-axis}. \end{cases}$

Table 4.2 presents the number of iterations required by the CG algorithm to converge to the solution for the multilevel scheme and the DCG method.

Problem 3. Truncated multilevel scheme for Poisson equation. We present results for solving the Poisson equation with Dirichlet boundary conditions using the truncated multilevel scheme and compare it with the multilevel scheme. Table 4.3 presents

TABLE 4.3
Comparison of multilevel and truncated multilevel scheme for the Laplace equation with Dirichlet boundary conditions.

Mesh size	Truncated multilevel			Multilevel	
($\sqrt{n}$)	k_0	Iter	Time	Iter	Time
16	2	17	.11	19	.12
32	2	19	.60	25	.48
64	3	27	2.52	35	2.83
128	3	28	12.77	49	16.18

the iterations and time required for both methods and the number of levels used by the truncated multilevel scheme. It may be observed that the condition number of the truncated multilevel scheme $\kappa(P_2^T P_2) = O(\sqrt[4]{n})$ and depends only on the number of levels k_0 to which the mesh is coarsened.

The results presented in Tables 4.1 and 4.2 indicate that the estimate for the condition number in Theorem 3.3 is quite pessimistic and that $\kappa(P_2^T P_2) = O(\sqrt{n})$ instead of $O(n^{0.7925})$. Moreover, the multilevel scheme displays similar behavior in the case of problems with varying inhomogeneity which demonstrates the robustness of its preconditioning. It must also be noted that the time to compute the matrices P, Z, and D increases linearly with the size of the problem, indicating the feasibility of the scheme for large-scale problems. Finally, we observe that the truncated multilevel scheme can be used to improve convergence at the expense of memory and computation per iteration. Similar results have been obtained for finite element discretization of the domain using unstructured meshes.

4.2. Generalized Stokes problem. In this section, we present experiments for the lid-driven cavity problem on a unit square using the multilevel scheme. The domain is discretized by a mesh defined on pressure nodes. We compare the multilevel algorithm to the Uzawa method preconditioned by the matrix C, where

$$C^{-1} = \nu M_p^{-1} + \alpha T_p^{-1}$$

in which M_p and T_p are the mass matrix and the discrete Laplace matrix defined on the pressure mesh. In the case of finite difference approximation, T_p is an M-matrix. This preconditioner was first presented in [8] and appears to be the most effective for our problem. The CG algorithm preconditioned by incomplete Cholesky factorization was used for the inner system solve, as well as the application of T_p^{-1} in the preconditioner. The pressure mass matrix M_p was approximated by a lumped diagonal matrix. To obtain a fair comparison, the fill-in was restricted so that the amount of storage used was similar to the multilevel scheme.

One advantage of multilevel scheme over other methods is its ability to ensure that the discrete divergence of the velocity is of the order of machine precision, which is a direct consequence of an explicit representation of the null-divergence basis P_2. Therefore, the use of the CG method in the multilevel scheme is merely to reduce the residual of the momentum equation to an acceptable tolerance. In contrast, the Uzawa methods implicitly reduce the divergence residual at the outer level and the momentum residual at the inner level.

The nature of the linear system (1.4) depends greatly on the ratio α/ν which determines the characteristics of the matrix A. Therefore, the main focus of the experiments was the behavior of each algorithm for $\alpha/\nu \in [0, \infty)$.

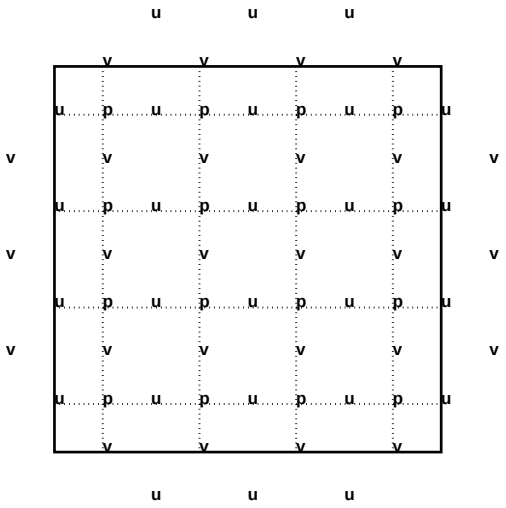

FIG. 4.1. *Finite difference method using the MAC discretization.*

TABLE 4.4
Comparison of Uzawa and multilevel for steady-state Stokes problem ($\alpha/\nu = 0$) discretized by the MAC scheme on a uniform mesh.

Size		Uzawa		Multilevel		
n_p	n_u	Iter	$time_U$	Iter	$time_{ML}$	$\frac{time_U}{time_{ML}}$
255	480	19 (39)	0.77	28	0.09	8.6
1023	1984	21 (72)	6.35	41	0.57	11.1
4095	8064	22 (140)	53.55	61	2.90	18.5
16383	32512	25 (272)	496.18	81	15.08	32.9

Problem 4. Finite difference method. We use the marker-and-cell (MAC) discretization scheme [15] on a staggered mesh (see Fig. 4.1). The velocity $\mathbf{u} = (0, 0)^T$ on the boundary except for the top edge where $\mathbf{u} = (1, 0)^T$, and $p = 0$ at the lower left corner to ensure full rank of B. Points on the boundary satisfy the boundary conditions, whereas points outside the domain must be interpolated. A tolerance of 10^{-9} was placed on the divergence residual.

Tables 4.4 and 4.5 present a comparison between the multilevel scheme and the Uzawa method for the extreme values of α/ν. The number of velocity and pressure unknowns are shown as n_u and n_p, respectively, and the numbers in parentheses give the total number of iterations at the inner level of the Uzawa method. Figure 4.2 shows that even in the worst case, the condition number of the preconditioned system grows as $O(\sqrt{n_p})$.

Problem 5. Finite elements method. In this case, we used an unstructured triangular mesh to discretize the domain. We chose the P1-isoP1 piecewise linear finite element pair for the mixed finite elements approximation, in which pressure is approximated by continuous piecewise linear elements on a triangular mesh $\mathcal{T}_h$, and velocity is approximated by continuous piecewise linear elements on a finer mesh $\mathcal{T}_{h/2}$ obtained by refining each element in the pressure mesh into four elements using the midpoints of each side.

TABLE 4.5
Comparison of Uzawa and multilevel for the generalized Stokes problem ($\alpha/\nu = 10^{10}$) discretized by the MAC scheme on a uniform mesh.

Size		Uzawa		Multilevel		
n_p	n_u	Iter	$time_U$	Iter	$time_{ML}$	$\frac{time_U}{time_{ML}}$
255	480	2 (22)	0.06	19	0.08	0.8
1023	1984	2 (40)	0.37	28	0.45	0.8
4095	8064	2 (75)	2.77	40	2.27	1.2
16383	32512	2 (144)	21.63	56	11.59	1.9

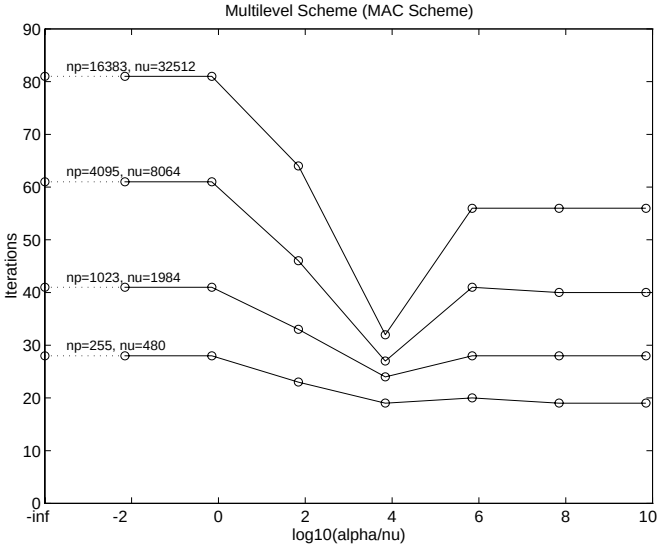

FIG. 4.2. *Iterations required by the multilevel scheme for $\alpha/\nu \in [0, 10^{10}]$ for the MAC discretization.*

TABLE 4.6
Comparison of Uzawa and multilevel for steady-state Stokes problem ($\alpha/\nu = 0$) using $P1$-iso$P1$ mixed finite elements.

Size		Uzawa		Multilevel		
n_p	n_u	Iter	$time_U$	Iter	$time_{ML}$	$\frac{time_U}{time_{ML}}$
351	2426	60 (72)	16.55	446	3.68	4.5
1334	9906	65 (139)	143.70	1077	36.97	3.9
2636	19938	67 (196)	413.57	1874	132.13	3.1
5178	39890	64 (274)	1237.50	3000	442.03	2.8

The velocity obeys Dirichlet boundary condition, and the pressure at the lower left corner is set to zero. In this case, a tolerance of 10^{-13} was enforced on the divergence residual. Tables 4.6 and 4.7 present a comparison between the multilevel scheme and the Uzawa method for the extreme values of α/ν. The relative behavior of these methods for other values of α/ν was observed to be an interpolation between these extremes.

TABLE 4.7
Comparison of Uzawa and multilevel for the generalized Stokes problem ($\alpha/\nu = 10^7$) using $P1$-iso$P1$ mixed finite elements.

Size		Uzawa		Multilevel		
n_p	n_u	Iter	$time_U$	Iter	$time_{ML}$	$\frac{time_U}{time_{ML}}$
351	2426	23 (19)	1.90	96	0.94	2.0
1334	9906	24 (25)	10.20	125	5.02	2.0
2636	19938	23 (30)	24.55	165	13.06	1.9
5178	39890	23 (38)	59.24	190	31.06	1.9

For our problems, the multilevel scheme is faster than the preconditioned Uzawa method for both the finite difference and finite element approximations. Moreover, our approach retains this edge for a wide choice of parameters α and ν which makes it a very effective method for these problems. It is interesting to note that for the MAC discretization, the preconditioning effect of the multilevel scheme is relatively independent of α/ν. Even though this property is not evident in the experiments with mixed finite element discretization, our approach appears to be very competitive with the Uzawa method.

The performance of the Uzawa method can be improved for uniform discretizations by the use of fast algorithms for inner system solves. However, for the more general case of unstructured meshes, the effectiveness of the Uzawa method depends greatly on the methods used for the solution of the inner systems. In this respect, we believe that our experiments demonstrate that the proposed multilevel scheme is very competitive for the solution of the generalized Stokes problem.

5. Conclusions. We have proposed a novel technique to solve the generalized Stokes problem, which computes a well-conditioned basis for the null-divergence velocity space and solves a reduced symmetric and positive definite system in that space using the CG method. We also describe a multilevel scheme for computing such a basis along with analysis of its convergence properties for the Poisson problem on a uniform mesh. Our approach is directly applicable to several discretization methods for self-adjoint elliptic problems including finite difference, finite element, and finite volume techniques. The extension of the multilevel scheme to convection-diffusion-type problems remains a topic for future research. We supplement our method with results of numerical experiments for finite difference and mixed finite element discretizations of the self-adjoint problems as well as the generalized Stokes problem. These experiments demonstrate that our approach provides a robust and effective preconditioning technique for these problems.

Appendix.

Algorithm Multilevel.

1. Initialize $E^{(0)} = E$, $i = 1$.
2. Divide the nodes in the mesh into l partitions.
3. If $l = 1$, then set $k = i$, $Z^{(k)} = V^{(k)}$, and

$$P^{(k)^T} = \begin{pmatrix} U^{(k)^T} & \\ & I \end{pmatrix},$$

where $E^{(i-1)} = U^{(i)} S^{(i)} V^{(i)^T}$. Go to step 9.

4. Compute the SVD $E_{int}^{(i-1)} = U^{(i)} S^{(i)} {V^{(i)}}^T$, where

$$E^{(i-1)} = \begin{pmatrix} E_{int}^{(i-1)} \\ E_{bnd}^{(i-1)} \end{pmatrix}$$

in which E_{int} and E_{bnd} consist of the rows corresponding to interior and boundary edges, respectively.
5. Compute $E_{bnd}^{(i-1)} \hat{V}^{(i)}$ and $Q^{(i)}$ to combine rows between neighboring active nodes in the coarse mesh.
6. Set $Z^{(i)} = V^{(i)}$ and

$${P^{(i)}}^T = \begin{pmatrix} I & \\ & {Q^{(i)}}^T \end{pmatrix} \begin{pmatrix} I & \\ -E_{bnd}^{(i-1)} V^{(i)} {S^{(i)}}^{\dagger} & I \end{pmatrix} \begin{pmatrix} {U^{(i)}}^T & \\ & I \end{pmatrix}.$$

7. Compute the gradient operator for the coarse mesh

$${Q^{(i)}}^T E_{bnd}^{(i-1)} \hat{V}^{(i)} = \begin{pmatrix} E^{(i)} \\ 0 \end{pmatrix}.$$

8. Set $i = i + 1$ and go to step 2.
9. Set $P = P^{(1)} P^{(2)} \cdots P^{(k)}$ and $Z = Z^{(1)} Z^{(2)} \cdots Z^{(k)}$.

REFERENCES

[1] O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, London, 1994.
[2] O. Axelsson and P. Vassilevski, *Algebraic multilevel preconditioning methods* 1, Numer. Math., 56 (1989), pp. 157–177.
[3] O. Axelsson and P. Vassilevski, *Algebraic multilevel preconditioning methods* 2, SIAM J. Numer. Anal., 27 (1990), pp. 1569–1590.
[4] R.E. Bank, B.D. Welfert, and H. Yserentant, *A class of iterative methods for solving saddle point problems*, Numer. Math., 56 (1990), pp. 645–666.
[5] J. H. Bramble and J. E. Pasciak, *Iterative Techniques for Time Dependent Stokes Problems*, Tech. report BNL-49970, Brookhaven National Laboratory, Upton, NY, 1994.
[6] R. Bramley, *An Orthogonal Projection Algorithm for Generalized Stokes Problems*, Tech. report 1190, CSRD, University of Illinois Urbana-Champaign, IL, 1992.
[7] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer Ser. Comput. Math., Springer-Verlag, Berlin, New York, 1991.
[8] J. Cahouet and J.-P. Chabard, *Some fast* 3*d finite element solvers for the generalized Stokes problem*, Internat. J. Numer. Methods Fluids, 8 (1988), pp. 869–895.
[9] N. Dyn and W.E. Ferguson, Jr., *The numerical solution of equality-constrained quadratic programming problems*, Math. Comp., 41 (1983), pp. 165–170.
[10] H. C. Elman and G. H. Golub, *Inexact and Preconditioned Uzawa Algorithms for Saddle Point Problems*, Tech. report CS-TR-3075, University of Maryland, College Park, MD, 1993.
[11] M. Fortin and R. Glowinski, *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*, North–Holland, New York, 1983.
[12] M. D. Gunzburger, *Finite Element Methods for Viscous Incompressible Flows*, Academic Press, New York, 1989.
[13] K. Gustafson and R. Hartman, *Divergence-free bases for finite element schemes in hydrodynamics*, SIAM J. Numer. Anal., 20 (1983), pp. 697–721.
[14] C. A. Hall, J. S. Peterson, T. A. Porsching, and F. R. Sledge, *The dual variable method for finite element discretizations of Navier-Stokes equations*, Internat. J. Numer. Methods Engrg., 21 (1985), pp. 883–898.
[15] R. Peyret and T. Taylor, *Computational Methods for Fluid Flows*, Springer Ser. Comput. Phys., Springer-Verlag, Berlin, New York, 1983.
[16] T. Rusten and R. Winther, *A preconditioned iterative method for saddle-point problems*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 887–904.

[17] A. Sameh and J. A. Wisniewski, *A trace minimization algorithm for the generalized eigenvalue problem*, SIAM J. Numer. Anal., 19 (1982), pp. 1243–1259.
[18] D. Silvester and A. Wathen, *Fast iterative solution of stabilised Stokes systems part* II: *Using general block preconditioners*, SIAM J. Numer. Anal., 31 (1994), pp. 1352–1367.
[19] R. Verfurth, *A combined conjugate gradient-multigrid algorithm for the numerical solution of the Stokes problem*, IMA J. Numer. Anal., 4 (1984), pp. 441–455.
[20] A. Wathen and D. Silvester, *Fast iterative solution of stabilised Stokes systems part* 1: *Using simple diagonal preconditioners*, SIAM J. Numer. Anal., 30 (1993), pp. 630–649.
[21] H. Yserentant, *On the multi-level splitting of finite element spaces*, Numer. Math., 49 (1986), pp. 379–412.