MULTIGRID AND KRYLOV SUBSPACE METHODS FOR THE DISCRETE STOKES EQUATIONS

Howard C. Elman*
Department of Computer Science and
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742
elman@cs.umd.edu

Report CS-TR-3302
UMIACS-TR-94-76
June 1994

Abstract. Discretization of the Stokes equations produces a symmetric indefinite system of linear equations. For stable discretizations, a variety of numerical methods have been proposed that have rates of convergence independent of the mesh size used in the discretization. In this paper, we compare the performance of four such methods: variants of the Uzawa, preconditioned conjugate gradient, preconditioned conjugate residual, and multigrid methods, for solving several two-dimensional model problems. The results indicate that where it is applicable, multigrid with smoothing based on incomplete factorizaton is more efficient than the other methods, but typically by no more than a factor of two. The conjugate residual method has the advantages of being both independent of iteration parameters and widely applicable.

Key words. Stokes, multigrid, Krylov subspace, conjugate gradient, conjugate residual, Uzawa

* This work was supported by the U. S. Army Research Office under grant DAAL-0392-G-0016, and by the National Science Foundation under grant ASC-8958544.

1. Introduction. Consider the system of partial differential equations

$$
\begin{array}{ll}
-\Delta u + \nabla p = f & \\
-\mathrm{div}\, u = 0 & \text{on } \Omega \\
u = 0 & \text{on } \partial\Omega, \\
\int_\Omega p = 0 &
\end{array}
\tag{1}
$$

where Ω is a simply connected bounded domain in $\mathbf{R}^d$, $d = 2$ or 3. This system, the *Stokes equations*, is a fundamental problem arising in computational fluid dynamics, see e.g. [7, 12, 14, 17]; u is the d-dimensional velocity vector defined on Ω, and p represents pressure.

Discretization of (1) by finite difference or finite element techniques leads to a linear system of equations of the form

$$
\begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix},
\tag{2}
$$

where A is a set of uncoupled discrete Laplacian operators and C is a positive semidefinite matrix. We consider here only *stable* discretizations, i.e., those for which the condition number of the Schur complement matrix $BA^{-1}B^T + C$ is bounded independently of the mesh size used in the discretization. For finite element discretizations with $C = 0$, this is a consequence of the *inf-sup condition* and upper bound

$$
\gamma \le \inf_q \sup_v \frac{(q, \mathrm{div}\, v)}{|v|_1 \, \|q\|_0}, \qquad \frac{|(q, \mathrm{div}\, v)|}{|v|_1 \, \|q\|_0} \le \Gamma,
$$

where γ and Γ are independent of the mesh size. Here, $|\cdot|_1$ and $\|\cdot\|_0$ denote the H^1-seminorm and Euclidean norm, respectively, on the discrete velocity and pressure spaces, and the bounds are taken over all v and q in the appropriate discrete spaces; see [7, 12, 14, 17].

In recent years, a variety of iterative algorithms have been devised for solving the discrete Stokes equations. In this paper, we compare the performance of four such methods:

1. a variant of the Uzawa method;
2. a preconditioned conjugate gradient (PCG) method applied to a transformed version of (2);
3. a preconditioned conjugate residual (PCR) method;
4. multigrid (MG).

The Uzawa method is the first among these to have been devised [2] and it is often advocated as an efficient solution technique, see e.g. [7, 12, 14]. The convergence factor associated with it is proportional to $(\kappa - 1)/(\kappa + 1)$ where κ is the condition number of the Schur complement $BA^{-1}B^T + C$ (see §2.5). The conjugate gradient method, developed by Bramble and Pasciak [5], has convergence factor proportional to $(\sqrt{\kappa}-1)/(\sqrt{\kappa}+1)$ but larger cost per step than the Uzawa method. The preconditioned conjugate residual method was developed by Rusten and Winther [24] and Silvester and Wathen [26, 31], and its convergence behavior is determined by properties of the indefinite matrix. For multigrid, we consider versions derived from two smoothing strategies: a variant of the distributed Gauss-Seidel method of Brandt and Dinar [6], and the technique based on incomplete factorization developed by Wittum [35].

These methods all have the property that for appropriate choice of preconditioners (or for multigrid, smoothers), their convergence rates are independent of the mesh size used in the discretization. The actual costs of using them depends on both the convergence rate and the cost per iteration. Our goal in this paper is to compare costs, in operation counts, of using each of the methods to solve three discrete versions of (1). For convergence to be independent of mesh size, the first three methods (*Krylov subspace methods*) require a preconditioning operator spectrally equivalent to the discrete Laplacian. In an effort to unify the comparison of these ideas with multigrid, we also implement this preconditioner using a multigrid method for the associated Poisson equation. Our main conclusions are as follows. For problems where it is applicable, one version of multigrid, using incomplete factorization, requires the fewest iterations and operations, but it is only marginally faster, i.e., by factors of approximately 1.5 to 2, than the Krylov subspace methods and the distributed Gauss-Seidel method. The Krylov subspace methods are more widely applicable than either multigrid method. Among the Krylov subspace methods, the conjugate residual method is slightly slower than the conjugate gradient method, and in some cases, the Uzawa method, but it has the advantage of not requiring any parameter estimates.

An outline of the rest of the paper is as follows. In §2, we present the solution algorithms and give an overview of their convergence properties. In §3, we specify four benchmark problems and the computational costs per iteration of each of the solution methods. In §4, we present the numerical comparison.

2. Overview of methods. In this section, we present the four algorithms under consideration and outline their convergence properties. The first three methods depend on a preconditioning operator Q_A that approximates the matrix A of (2). We assume that Q_A is symmetric positive definite (SPD) and that

$$\eta_1 \leq \frac{(v, Av)}{(v, Q_A v)} \leq \eta_2 \,, \tag{3}$$

where η_1 and η_2 are independent of the mesh size used in the discretization. In addition, finite element discretizations of (1) have a mass matrix M associated with the pressure discretization.[1] The preconditioner will also include a SPD approximation Q_M of M. Discussions of computational costs will be made in terms of various matrix operations together with inner products and "AXPY's," i.e., vector operations of the form $y \leftarrow \alpha x + y$.

2.1. The inexact Uzawa method. We use the following "inexact" version of the Uzawa algorithm [11], which starts with $u_0 \equiv 0$ and an arbitrary initial guess p_0:

$$\begin{aligned} &\texttt{for } i = 0 \texttt{ until} \text{ convergence, } \texttt{do} \\ &\quad u_{i+1} = u_i + Q_A^{-1}(f - (Au_i + B^T p_i)) \\ &\quad p_{i+1} = p_i + \alpha\, Q_M^{-1}(Bu_{i+1} - Cp_i) \\ &\texttt{enddo} \end{aligned} \tag{4}$$

Here, α is a scalar parameter that must be determined prior to the iteration.

In the "exact" version of this algorithm, $Q_A = A$ and the first step is equivalent to solving the linear system $Au_{i+1} = f - B^T p_i$. When $Q_M = I$, the exact algorithm

[1] If the finite element solution is expressed using a given basis $\{\phi_i\}$ as $p = \sum_i \delta_i \phi_i$, then $\|p\|_{L_2} = (\delta, M\delta)^{1/2}$.

is then a fixed parameter first order Richardson iteration applied to the Schur complement system $(BA^{-1}B^T + C)p = BA^{-1}f$; Q_M is a preconditioner for this iteration. The inexact Uzawa algorithm (4) replaces the exact computation of $A^{-1}(f - B^T p_i)$ with an approximation.

2.2. A preconditioned conjugate gradient method. Let $\mathcal{A}$ denote the coefficient matrix of (2). Premultiplication of (2) by the matrix

$$\mathcal{T} = \begin{pmatrix} Q_A^{-1} & 0 \\ BQ_A^{-1} & -I \end{pmatrix}$$

produces the equivalent system

$$\begin{pmatrix} Q_A^{-1}A & Q_A^{-1}B^T \\ BQ_A^{-1}A - B & BQ_A^{-1}B^T + C \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} Q_A^{-1}f \\ BQ_A^{-1}f \end{pmatrix}. \tag{5}$$

Let $\mathcal{M} = \mathcal{T}\mathcal{A}$ denote the coefficient matrix of this system. The conjugate gradient method (CG) developed in [5] requires that the bilinear form

$$\left[\begin{pmatrix} v_1 \\ q_1 \end{pmatrix}, \begin{pmatrix} v_2 \\ q_2 \end{pmatrix} \right] \equiv ((A - Q_A)v_1, v_2) + (q_1, q_2) \tag{6}$$

define an inner product. Equivalently, the preconditioning operator Q_A must satisfy (3) with $\eta_1 > 1$. It is shown in [5] that $\mathcal{M}$ is SPD with respect to the inner product (6), so that CG in this inner product is applicable. The matrix

$$\mathcal{G} = \begin{pmatrix} I & 0 \\ 0 & Q_M \end{pmatrix}. \tag{7}$$

is also SPD with respect to (6), so that this can be used as a preconditioner.

Let

$$X_0 = \begin{pmatrix} u_0 \\ p_0 \end{pmatrix}, \quad R_0 = \begin{pmatrix} f - (Au_0 + B^T p_0) \\ -(Bu_0 - Cp_0) \end{pmatrix}$$

denote an arbitrary guess for the solution and the associated residual. An implementation of PCG is given below. Except for the nonstandard inner product, it is the standard implementation, as given for example in [15, p. 529]. It is more efficient then the version given in [5]. The preconditioner Q_A is implicitly incorporated into the inner product. The use of a preconditioner (7) is new.

$$
\begin{aligned}
&\hat{R}_0 = \mathcal{T} R_0, \qquad \tilde{R}_0 = \mathcal{G}^{-1}\hat{R}_0\\
&P_0 = \tilde{R}_0, \qquad \mathcal{M}P_0 = \mathcal{T}\mathcal{A}P_0\\
&\alpha_0^{(n)} = [\hat{R}_0, \tilde{R}_0], \qquad \alpha_0^{(d)} = [P_0, \mathcal{M}P_0], \qquad \alpha_0 = \alpha_0^{(n)}/\alpha_0^{(d)}\\
&X_1 = X_0 + \alpha_0 P_0\\
&R_1 = R_0 - \alpha_0 \mathcal{A} P_0, \qquad \hat{R}_1 = \hat{R}_0 - \alpha_0 \mathcal{M} P_0, \qquad \tilde{R}_1 = \mathcal{G}^{-1}\hat{R}_1\\
&\texttt{for } i = 1 \texttt{ until} \text{ convergence, } \texttt{do}\\
&\quad \beta_{i-1}^{(n)} = [\hat{R}_i, \tilde{R}_i], \qquad \beta_{i-1}^{(d)} = \alpha_{i-1}^{(n)}, \qquad \beta_{i-1} = \beta_{i-1}^{(n)}/\beta_{i-1}^{(d)}\\
&\quad P_i = \tilde{R}_i + \beta_{i-1}P_{i-1}, \qquad \mathcal{M}P_i = \mathcal{T}\mathcal{A}P_i\\
&\quad \alpha_i^{(n)} = \beta_{i-1}^{(n)}, \qquad \alpha_i^{(d)} = [P_i, \mathcal{M}P_i], \qquad \alpha_i = \alpha_i^{(n)}/\alpha_i^{(d)}\\
&\quad X_{i+1} = X_i + \alpha_i P_i, \qquad R_{i+1} = R_i - \alpha_i \mathcal{A}P_i\\
&\quad \hat{R}_{i+1} = \hat{R}_i - \alpha_i \mathcal{M}P_i, \qquad \tilde{R}_{i+1} = \mathcal{G}^{-1}\hat{R}_{i+1}\\
&\texttt{enddo}
\end{aligned}
$$

To help identify operation counts, we describe the computation of $\{\alpha_i\}$ and $\{\beta_i\}$ in more detail. Letting

$$
R_i = \begin{pmatrix} r_i \\ s_i \end{pmatrix}, \ \hat{R}_i = \begin{pmatrix} \hat{r}_i \\ \hat{s}_i \end{pmatrix}, \ \tilde{R}_i = \begin{pmatrix} \hat{r}_i \\ \tilde{s}_i \end{pmatrix},
$$

we have $\beta_{i-1}^{(n)} = [\hat{R}_i, \tilde{R}_i] = (\hat{r}_i, A\hat{r}_i - r_i) + (\hat{s}_i, \tilde{s}_i)$; similarly, if

$$
P_i = \begin{pmatrix} c_i \\ d_i \end{pmatrix}, \ \mathcal{A}P_i = \begin{pmatrix} v_i \\ w_i \end{pmatrix}, \ \mathcal{M}P_i = \begin{pmatrix} Q_A^{-1} v_i \\ BQ_A^{-1} v_i - w_i \end{pmatrix}, \tag{8}
$$

then $\alpha_i^{(d)} = [P_i, \mathcal{M}P_i] = (c_i, AQ_A^{-1}v_i - v_i) + (d_i, BQ_A^{-1}v_i - w_i)$. Q_A is referenced only in the construction of $Q_A^{-1}v$ in (8), so that only the action of the inverse of Q_A is required. Moreover, although the vectors $A\hat{r}_i$, Ac_i (for v_i) and $AQ_A^{-1}v_i$ are used, the first two of these can be computed using an AXPY. Consequently, only one matrix-vector product by A is needed.

2.3. The preconditioned conjugate residual method. Since $\mathcal{A}$ is symmetric, variants of the conjugate residual method are applicable. Let X_0 denote the initial guess and R_0 its residual. The following algorithm implements the ORTHOMIN version of PCR with preconditioner $\mathcal{Q}$ [3].[2]

$$
\begin{array}{l}
\tilde{R}_0 = \mathcal{Q}^{-1}R_0,\ P_0 = \tilde{R}_0,\ S_0 = \mathcal{Q}^{-1}\mathcal{A}P_0 \\
\alpha_0^{(n)} = (\tilde{R}_0, \mathcal{A}P_0),\ \alpha_0^{(d)} = (\mathcal{A}P_0, S_0),\ \alpha_0 = \alpha_0^{(n)}/\alpha_0^{(d)} \\
X_1 = X_0 + \alpha_0 P_0,\ R_1 = R_0 - \alpha_0\mathcal{A}P_0,\ \tilde{R}_1 = \tilde{R}_0 - \alpha_0 S_0 \\
\texttt{for } i = 1 \texttt{ until} \text{ convergence, } \texttt{do} \\
\quad \beta_{i-1}^{(n)} = -(\mathcal{A}\tilde{R}_i, S_{i-1}), \qquad \beta_{i-1}^{(d)} = \alpha_{i-1}^{(d)} \\
\quad P_i = \tilde{R}_i + \beta_{i-1}P_{i-1}, \qquad \mathcal{A}P_i = \mathcal{A}\tilde{R}_i + \beta_{i-1}\mathcal{A}P_{i-1}, \qquad S_i = \mathcal{Q}^{-1}\mathcal{A}P_i \\
\quad \alpha_i^{(n)} = (\tilde{R}_i, \mathcal{A}P_i), \qquad \alpha_i^{(d)} = (\mathcal{A}P_i, S_i), \qquad \alpha_i = \alpha_i^{(n)}/\alpha_i^{(d)} \\
\quad X_{i+1} = X_i + \alpha_i P_i, \qquad R_{i+1} = R_i - \alpha_i\mathcal{A}P_i, \qquad \tilde{R}_{i+1} = \tilde{R}_i - \alpha_i S_i \\
\texttt{enddo}
\end{array}
$$

Any symmetric positive-definite $\mathcal{Q}$ could be used as a preconditioner. As in [26], we use

$$
\mathcal{Q} = \begin{pmatrix} Q_A & 0 \\ 0 & Q_M \end{pmatrix}.
$$

2.4. Multigrid. As is well known, multigrid methods combine iterative methods to smooth the error with correction derived from a coarse grid computation. We use V-cycle multigrid for "transformed systems." Our description follows [34, 35]. Cf. [22, 30] for other multigrid methods derived from the squared system associated with (2).

Let $-\Delta_p$ denote the Laplace operator defined on the pressure space, with Neumann boundary conditions (see [16]), and let A_p be a discrete approximation to $-\Delta_p$ defined on the pressure grid. Consider the following transformed version of (2):

$$
\begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} I & B^T \\ 0 & -A_p \end{pmatrix} \begin{pmatrix} \hat{u} \\ \hat{p} \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}, \qquad \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} I & B^T \\ 0 & -A_p \end{pmatrix} \begin{pmatrix} \hat{u} \\ \hat{p} \end{pmatrix}. \tag{9}
$$

The coefficient matrix in (9) is

$$
\tilde{\mathcal{A}} = \begin{pmatrix} A & W \\ B & G \end{pmatrix}, \tag{10}
$$

where $W = AB^T - B^T A_p$ and $G = BB^T + CA_p$. For appropriate discretizations of (1) (see §3), W is of low rank, with nonzero entries only in rows corresponding to mesh points next to $\partial\Omega$. When $C = 0$, G can also be viewed as discretization of $-\Delta_p$. The splitting

$$
\tilde{\mathcal{A}} = \mathcal{S} - \mathcal{R} \tag{11}
$$

then induces a stationary iteration applicable to (2),

$$
\begin{pmatrix} u_{k+1} \\ p_{k+1} \end{pmatrix} = \begin{pmatrix} u_k \\ p_k \end{pmatrix} + \begin{pmatrix} I & B^T \\ 0 & -A_p \end{pmatrix} \mathcal{S}^{-1} \begin{pmatrix} f - (Au_k + B^T p_k) \\ -(Bu_k - Cp_k) \end{pmatrix}. \tag{12}
$$

[2] It is possible for this version of PCR to break down, with $\alpha_i = 0$. The ORTHODIR version, which uses a three-term recurrence to generate P_i, is guaranteed not to break down; it requires two additional AXPY's. Our implementation switches from the ORTHOMIN to ORTHODIR direction update if $|\alpha_i| < 10^{-4}$, as described in [9]. In the experiments discussed in §4, this switch never took place.

This is used as the smoother for the multigrid solver for (2). Specific choices for $\mathcal{S}$ are given in §3.2.

Let R_u denote a restriction operator mapping velocity vectors in the fine grid (of width h) to the coarse grid (of width $2h$), let R_p similarly denote the restriction operator for the discrete pressure space, and let P_u and P_p denote prolongation operators from the coarse spaces to the fine spaces. (For simplicity, we are omitting explicit mention of h in this notation.) One step of V-cycle multigrid for solving (2), starting with initial guess u^0, p^0, is as follows.

$(u^1, p^1) = \mathbf{MG}(u^0, p^0, f, g, k_1, k_2, h)$

`if` $h < h_0$, `then` % Recursive call

Starting with u^0, p^0, perform k_1 smoothing steps (12), producing $u^{1/3}, p^{1/3}$

$r^{1/3} = f - (Au^{1/3} + B^T p^{1/3}), \qquad s^{1/3} = -(Bu^{1/3} - Cp^{1/3})$

$r_c^{1/3} = R_u r^{1/3}, \qquad s_c^{1/3} = R_p s^{1/3}$

$(u_c^{2/3}, p_c^{2/3}) = \mathbf{MG}(0, 0, r_c^{1/3}, s_c^{1/3}, k_1, k_2, 2h)$

$u^{2/3} = u^{1/3} + P_u u_c^{2/3}, \qquad p^{2/3} = p^{1/3} + P_p p_c^{2/3}$

Starting with $u^{2/3}, p^{2/3}$, perform k_2 smoothing steps (12), producing u^1, p^1

`else` % Coarse grid solve when $h = h_0$

Solve $\begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} u^1 \\ p^1 \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}$ directly

`endif`

We also use V-cycle multigrid derived from the discrete Laplacian as a preconditioner to approximate the action of A^{-1} for the Krylov subspace methods; this is defined analogously and we omit the details. For all multigrid methods, we use bilinear interpolation to define P_u and P_p, and $R_u = P_u^T$, $R_p = P_p^T$. The discrete operators at each level are derived from the discretization on the associated grid.

2.5. Convergence properties. We briefly outline some convergence properties of these methods; see the primary references for derivations of bounds. Each of the methods generates a sequence of iterates $u_i \approx u$, $p_i \approx p$ such that, if e_i is a representation of the error, then $\lim_{i\to\infty}(\|e_i\|/\|e_0\|)^{1/i} = \rho$ for some norm $\|\cdot\|$. We refer to ρ as the *convergence factor*.

We are assuming that the discretization and choice of Q_M are such that

$$\lambda_1 \le \frac{(q, (BA^{-1}B^T + C)q)}{(q, Q_M q)} \le \lambda_2, \tag{13}$$

where λ_1 and λ_2, and therefore, $\kappa \equiv \lambda_2/\lambda_2$, are bounded independently of the mesh size of the discretization. This is the case, for example, when Q_M is a suitable approximation of the mass matrix in finite element discretization [29, 32]. Note that κ is the spectral condition number of $Q_M^{-1}(BA^{-1}B^T + C)$.

The exact Uzawa algorithm has convergence factor $\rho\left(I - \alpha\, Q_M^{-1}(BA^{-1}B^T + C)\right)$ [12]. This is smallest for the choice $\alpha = 2/(\lambda_1 + \lambda_2)$, in which case it has the value $(\kappa - 1)/(\kappa + 1)$. Thus, the convergence factor for the Uzawa algorithm is independent of the mesh. It is shown in [11] that the performance of the inexact Uzawa algorithm is close to that of the exact one if the iterate u_{i+1} satisfies

$$\|f - B^T p_i - A u_{i+1}\|_2 < \tau \|B u_i - C p_i\|_{Q_A^{-1}}, \tag{14}$$

where τ is independent of the mesh size.

The PCG method is analyzed in [5, Theorem 1], where it is shown that the condition number of the coefficient matrix $\mathcal{M}$ of (5) is bounded by a constant proportional to κ. Thus, standard results for CG [15] imply that the bound on the convergence factor for this method is proportional to $(\sqrt{\kappa}-1)/(\sqrt{\kappa}+1)$. The constant of proportionality depends on how close η_1 is to η_2 in (3), i.e., how well Q_A approximates A.

The PCR method is analyzed in [24, 26]. The analysis shows that the eigenvalues of the preconditioned matrix $\mathcal{Q}^{-1}\mathcal{A}$ are contained in two intervals $[-a,-b] \cup [c,d]$, where a, b, c, are d are positive constants that are independent of the mesh size. The sizes of the intervals depend on κ and the accuracy with which Q_A approximates A. It follows from the convergence analysis of CR [9, 27] that the convergence factor for the preconditioned algorithm is independent of the mesh size. For example, it is shown [9] that if $d-c = a-b > 0$, then the convergence factor is bounded by $2\left(\frac{1-\sqrt{\beta}}{1+\sqrt{\beta}}\right)^{1/2}$, where $\beta = (bc)/(ad)$.

It is shown in [36] that for finite difference discretization of (1) (see §3.1), two-grid variants of multigrid are convergent with convergence rate independent of the mesh size. The analysis applies to the ILU smoothing of §3.2, although it requires that the prolongation be based on biquadratic interpolation. In practice, bilinear interpolation has been observed to be sufficient [35]. Fourier analysis in [6] also suggests that MG/DGS has convergence rate independent of mesh size.

Remark 2.1. Several other proposed methods share properties with the version of PCG under consideration. In particular, Verfürth [29] has shown that PCG applied directly to the Schur complement system has convergence factor proportional to ρ_{CG}; however, this method requires accurate computation of the action of A^{-1} at each CG step [23]. Bank, Welfert and Yserentant [4] present a method making use of $Q_A \approx A$ with convergence rate dependent on the accuracy of this approximation, but using an additional inner iteration on the pressure space.

3. Solution costs.

In this section, we outline the computational costs required to solve three benchmark problems on $\Omega = (0,1)\times(0,1)$, for each of the solution methods of §2.

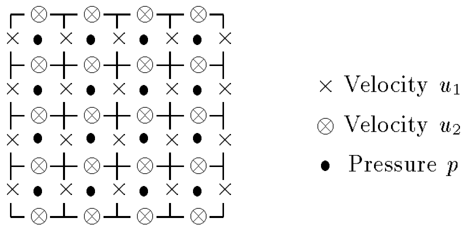
3.1. Benchmark problems.

We use four discretizations to produce test problems: "marker and cell" finite differences, and three mixed finite element strategies.

1. *Finite differences* [19]. This consists of the usual five-point operator for each of the discrete Laplacian operators of (1), together with centered differences for the first derivatives ∇p and $\operatorname{div} u$. For the discretization to be stable, it is necessary to use staggered grids in $\bar{\Omega}$. Figure 1 shows such grids on a mesh of width $h = 1/4$. In order to define the velocity discretizations at grid points next to $\partial\Omega$, certain values outside $\bar{\Omega}$ must be extrapolated; for example, this is needed to approximate $\partial^2 u_1/\partial y^2$ for points "$\times$" next to the bottom of $\partial\Omega$.

2. *Linear/constant finite elements.* This choice consists of continuous piecewise linear velocities on a mesh of width h, and piecewise constant pressures on a mesh of width

FIG. 1. *Staggered grids for finite difference discretization.*



$2h$. The discrete pressures are not required to be continuous. The coarser pressure grid ensures that the inf-sup condition holds [17]. We refer to this as the $P_1(h)P_0(2h)$ discretization.

3. *Piecewise linear finite elements.* Here, continuous piecewise linear velocities on a mesh of width h are paired with continuous piecewise linear pressures on a mesh of width $2h$. The inf-sup condition is also satisfied. We call this the $P_1(h)P_1(2h)$ discretization.

4. *Stabilized piecewise linear finite elements.* A stable discretization using piecewise linear velocities and pressures on a single of mesh can be obtained using a stabilization matrix $C = \beta h^2 A_n$, where A_n is the discrete Laplace operator defined on the pressure space, subject to Neumann boundary conditions [8]. This technique is equivalent to the mini-element discretization [1] after elimination of the internal degrees of freedom. We use $\beta = .025$, as recommended in [25]. We refer to this discretization as $P_1(h)P_1(h)$.

The usual hat functions are used as the bases for linear velocities and pressures.

The coefficient matrix $\mathcal{A}$ of (2) for all these problems, as well as B^T, C, and $BA^{-1}B^T + C$, are rank deficient by one; the latter three matrices share a constant null vector. As a result, the discrete pressure solutions are uniquely defined only up to a constant. In exact arithmetic, the solution methods under consideration correct the initial guess with quantities orthogonal to the null space of $\mathcal{A}$, so that the component of the null space in the computed solution is the same as in the initial guess. For the analysis, the lower bound of (13) refers to the smallest nonzero eigenvalue.

Note that our goal in considering these problems is to compare the performance of the different solution strategies on a variety of problems. We highlight some properties of each of the problems as follows:

1. finite differences, stable, #(pressure unknowns) $\approx$ #(velocity grid points);
2. finite elements, stable, discontinuous pressures, #(pressure unknowns) $\approx \frac{1}{2}$ #(velocity grid points);
3. finite elements, stable, continuous pressures, #(pressure unknowns) $\approx \frac{1}{4}$ #(velocity grid points);
4. finite elements, requires stabilization, continuous pressures, #(pressure unknowns) $\approx$ #(velocity grid points).

We are not comparing the accuracy achieved by the discretizations, and remark only that the three finite element discretizations display the same asymptotic convergence rates. See [17, pp. 29,50] for comments on accuracy of finite element discretization, and [21] for analysis of the finite difference scheme.

3.2. Preconditioners and smoothers. The Uzawa, PCR, and PCG methods require choices of Q_A and Q_M. For all cases, Q_A consists of one step of V-cycle multigrid derived from the discrete Laplacian. To ensure that Q_A is symmetric, the smoothing is based on damped point-Jacobi iteration with damping parameter $\omega = 2/3$ [20]. For the three finite element discretizations, Q_M is chosen to be the diagonal of the mass matrix M, see [32]. (In the case of the $P_1(h)P_0(2h)$ discretization, $Q_M = M$.) Although there is no mass matrix for finite differences, a natural analogue in two dimensions is $M = h^2 I$, and this is used for Q_M with finite differences.

We consider two multigrid smoothing strategies. The first is a variant of the distributed Gauss-Seidel (DGS) iteration introduced by Brandt and Dinar [6]. The splitting operator of (11) is given by

$$\mathcal{S} = \begin{pmatrix} S_A & 0 \\ B & S_G \end{pmatrix},$$

so that the smoother (12) has the form

$$\begin{aligned} \tilde{u}_{k+1} &= S_A^{-1}(f - (Au_k + B^T p_k)) \\ \tilde{p}_{k+1} &= S_G^{-1}(-(B(u_k + \tilde{u}_{k+1}) + Cp_k) \\ u_{k+1} &= u_k + \tilde{u}_{k+1} + B^T \tilde{p}_{k+1} \\ p_{k+1} &= p_k - A_p \tilde{p}_{k+1}\,. \end{aligned}$$

For S_A, we use the point Gauss-Seidel matrix derived from red-black ordering of the velocity grid. (That is, if $A = D - L - U$ with the red-black ordering, then $S_A = D - L$.) For finite differences, $S_G = (1/\omega)T$ where T is the tridiagonal part of G and $\omega = 2/3$; that is, S_G corresponds to a damped one-line Jacobi splitting. For $P_1(h)P_1(h)$ finite elements, S_G is the block Jacobi matrix derived from a two-line ordering of the underlying grid. These are slightly more sophisticated versions of the choice $S_G = diag(G)$ used in [6]. We refer to this multigrid method as MG/DGS.

The other multigrid smoother is the incomplete LU factorization (ILU) presented by Wittum [35]. We use an ILU factorization of the matrix $\tilde{\mathcal{A}}$ of (10), with no fill-in in the factors. The ordering for $\tilde{\mathcal{A}}$ is problem dependent. For finite differences, it is derived from an uncoupled red-black ordering of the underlying grid. That is, the grid values for u_1 were listed first, in red-black ordering, followed by those for u_2, and then those for p. (See also Remark 3.3 below.) For $P_1(h)P_1(h)$ finite elements, $\tilde{\mathcal{A}}$ is ordered according to an uncoupled *lexicographic* ordering of the grid vectors. We denote this method by MG/ILU.

In choosing preconditioners and smoothers, we have attempted to use methods that are suitable for vector and parallel computers. Thus, we are using point Jacobi smoothing for multigrid preconditioning, red-black Gauss-Seidel and line Jacobi for the DGS iteration, and a red-black ordering for MG/ILU applied to finite differences. With the $P_1(h)P_1(h)$ discretization, the operator G in the DGS method is a 19-point operator that has block Property A for a two-line ordering of the pressure grid, so that the two-line Jacobi splitting can be implemented efficiently in parallel. The ILU smoother used with this problem is not efficient on parallel computers. Our multigrid strategies do not address the issue of idleness of parallel processors for coarse grid computations; see [10, 13] for discussions of this point for the discrete Poisson equation.

Parameters are required for the Uzawa, PCG and multigrid methods, and for the multigrid preconditioner. These are as follows:

UZAWA: The optimal value of α for the exact Uzawa method, determined empirically, is used for the inexact version. This requires computation of the extreme eigenvalues of $Q_M^{-1}(BA^{-1}B^T + C)$.

PCG: As noted in §2.3, the preconditioner must be scaled so that $\eta_1 > 1$ in (3). From the results of [5], it is desirable to have η_1 close to 1. In all tests, the scaling is chosen so that $1 < \eta_1 < 1.02$. This requires computation of the smallest eigenvalue of $Q_A^{-1}A$.[3]

MULTIGRID: For the coarse mesh size h_0 in multigrid computations, we chose the one of $h_0 = 1/2$ and $h_0 = 1/4$ that produced lower iteration counts. This turned out to be $h_0 = 1/2$ for preconditioners and $h_0 = 1/4$ for solvers. The coarse grid solution is obtained using Cholesky factorization for the preconditioners and singular value decomposition for the solvers.

REMARK 3.1. For the Uzawa method, the choice of Q_A does not guarantee that the condition (14) is satisfied. The results of [11, 33] as well as those of §4 suggest that with multigrid for Q_A, (14) may be too stringent.

REMARK 3.2. The effectiveness of the multigrid solvers depends on the fact that the commutator W in (10) is zero away from the boundary of Ω. This is true for the finite difference and stabilized $P_1(h)P_1(h)$ discretizations, where pressures and velocities are defined on the same grid, but not for the (stable) $P_1(h)P_1(2h)$ discretization. Our experiments confirm that multigrid is ineffective for this discretization, and we do not include it as a option. See [18, p. 248] for a discussion of this issue. For the $P_1(h)P_0(2h)$ discretization, it is difficult to define the discrete pressure Poisson operator A_p, and we know of no multigrid implementation for this problem.

REMARK 3.3. For MG/ILU applied to the finite difference discretization, we also tested several alternative ordering strategies, including an uncoupled lexicographic ordering (i.e., like that used for $P_1(h)P_1(h)$), as well as several "coupled" lexicographic orderings. For the latter strategies, velocity and pressure unknowns are not separated from one another, see [28]. The performances of MG/ILU for all these orderings were very close. For example, for $h = 1/32$ as in Table 4 below, the smallest average iteration count with one smoothing step was $10\frac{1}{3}$ and the largest was $11\frac{2}{3}$.

3.3. Iteration costs. We identify the costs per iteration of each of the methods by first specifying the "high level" operations of which they are composed, and then determining the costs of each of these operations. High level operations are defined to be matrix-vector products, inner products (denoted "(,)" in the tables of this section), and AXPY's. Note that each of the techniques under consideration is formulated with essentially the same set of these operations; consequently, we expect operation counts to give a good idea of their comparative performance.

The high level operations are shown in Table 1. Matrix-vector products include operations with matrices that define the problem or method, such as A or R_u, as well as preconditioning and smoothing operators such as Q_A^{-1} and S_A^{-1}. The latter computations are themselves built from other matrix operations, and some of these are also identified in the table. All multigrid entries correspond to operations performed on one grid level. For multigrid solvers, the smoothing operations are presented separately; these operations would be performed k_1 times during presmoothing and k_2

[3] In the experiments described in §4, these were computed using a power method applied to $Q_A^{-1}A - I$; five to ten steps were needed to obtain an estimate accurate to three significant digits.

TABLE 1
High level operations for all solution algorithms.

	Matrix-Vector Product			AXPY	(,)
Uzawa	1 A	1 B^T	1 Q_A^{-1}	1 (n_p)	1 (n_u+n_p)
	1 B	1 C	1 Q_M^{-1}		
PCG	1 A	1 B^T	1 Q_A^{-1}	4 (n_u+n_p)	3 (n_u+n_p)
	2 B	1 C	1 Q_M^{-1}	2 (n_u)	
PCR	1 A	1 B^T	1 Q_A^{-1}	5 (n_u+n_p)	4 (n_u+n_p)
	1 B	1 C	1 Q_M^{-1}		
Multigrid Preconditioner	$(1+k_1+k_2)\,A$		1 R_u		
	$(k_1+k_2)\,S_A^{-1}$		1 P_u		
Multigrid Solver (Excluding smoother)	1 A	1 B^T	1 R_u		1 (n_u+n_p)
	1 B	1 C	1 R_p		
	1 P_u	1 P_p			
DGS Smoother	1 A	2 B^T	1 A_p		
	1 B	1 C	1 S_A^{-1}		
			1 S_G^{-1}		
ILU Smoother	1 A	2 B^T	1 A_p		
	1 B	1 C	1 $\mathcal{S}^{-1}$		

times during postsmoothing. The lengths of the vector operations are listed in parentheses. We are assuming that one inner product will be used in the convergence test, and the counts in the table include this.

The costs of matrix-vector products are estimated to be the number of nonzeros in the matrices used. This is roughly one half the number of "FLOPS" required, and it is also proportional to the number of memory references. These costs, for discretizations in which the velocity unknowns come from an $n \times n$ grid, are shown in Table 2. The costs of vector operations are taken to be the length of the vectors.

Combining the data of Tables 1 and 2 gives an estimate for the cost per iteration for each of the solution methods under consideration. These numbers are all proportional to n^2, and we present in Table 3 the cost factors obtained by omitting this factor, rounded to the nearest integer. For the multigrid methods (preconditioners and solvers), the cost of one full multigrid step is estimated as 4/3 times the cost of the computations on the finest grid; this is approximately the cost of full recursive multigrid in two dimensions.

4. Experimental results. We now present the results of numerical experiments for solving (2). All experiments were performed in MATLAB on a SPARC-10 workstation. For each solution algorithm, we solved three problems derived from three choices of f consisting of uniformly distributed random numbers in $[-1, 1]$. The initial guess in all cases was $u_0 = 0$, $p_0 = 0$. The stopping criterion was

$$\|R_i\|_2/\|R_0\|_2 < 10^{-6},$$

Table 2
Costs for matrix-vector products.

	Fin. Diff.	$P_1(h)P_0(2h)$	$P_1(h)P_1(2h)$	$P_1(h)P_1(h)$
A	$10n^2$	$10n^2$	$10n^2$	$10n^2$
B, B^T	$4n^2$	$4n^2$	$8n^2$	$12n^2$
C	0	0	0	$5n^2$
Q_M^{-1}	$1n^2$	$0.25n^2$	$0.25n^2$	$1n^2$
S_A^{-1} (Jacobi)	$2n^2$	$2n^2$	$2n^2$	$2n^2$
S_A^{-1} (Gauss-Seidel)	$6n^2$	$6n^2$	$6n^2$	$6n^2$
S_G^{-1}	$3n^2$	–	–	$9n^2$
A_p	$5n^2$	–	–	$5n^2$
R_u, P_u	$6n^2$	$4.5n^2$	$4.5n^2$	$4.5n^2$
R_p, P_p	$3n^2$	–	–	$2.25n^2$
$\mathcal{S}^{-1}$	$19n^2$	–	–	$41n^2$

Table 3
Cost factors.

		Uzawa	PCR	PCG	MG/DGS	MG/IC
Finite	$k_1 = k_2 = 1$	84	107	109	148	175
Differences	$k_1 = k_2 = 2$	116	139	141	244	297
$P_1(h)P_0(2h)$	$k_1 = k_2 = 1$	79	98	101	–	–
	$k_2 = k_2 = 2$	111	130	133	–	–
$P_1(h)P_1(2h)$	$k_1 = k_2 = 1$	86	104	111	–	–
	$k_2 = k_2 = 2$	118	136	143	–	–
$P_1(h)P_1(h)$	$k_1 = k_2 = 1$	101	124	134	247	333
	$k_2 = k_2 = 2$	133	156	166	421	591

Table 4
Iterations.

		Uzawa	PCR	PCG	MG/DGS	MG/ILU
Finite	$k_1 = k_2 = 1$	36	41	30	24	12
Differences	$k_1 = k_2 = 2$	28	33	23	15	9
$P_1(h)P_0(2h)$	$k_1 = k_2 = 1$	34	41	29	–	–
	$k_2 = k_2 = 2$	26	34	23	–	–
$P_1(h)P_1(2h)$	$k_1 = k_2 = 1$	89	57	38	–	–
	$k_2 = k_2 = 2$	89	50	31	–	–
$P_1(h)P_1(h)$	$k_1 = k_2 = 1$	39	47	32	20	8
	$k_1 = k_2 = 2$	38	40	25	10	7

Table 5
Estimates of convergence factors.

		Uzawa	PCR	PCG	MG/DGS	MG/ILU
Finite	$k_1 = k_2 = 1$	.67	.70	.66	.62	.39
Differences	$k_1 = k_2 = 2$	.60	.64	.57	.50	.31
$P_1(h)P_0(2h)$	$k_1 = k_2 = 1$	.69	.69	.70	–	–
	$k_2 = k_2 = 2$	.58	.66	.55	–	–
$P_1(h)P_1(2h)$	$k_1 = k_2 = 1$	.82	.79	.75	–	–
	$k_2 = k_2 = 2$	.84	.78	.70	–	–
$P_1(h)P_1(h)$	$k_1 = k_2 = 1$	.70	.75	.68	.56	.24
	$k_1 = k_2 = 2$	.70	.74	.62	.33	.21

where

$$R_i = \begin{pmatrix} f \\ 0 \end{pmatrix} - \begin{pmatrix} A & B^T \\ B & -C \end{pmatrix} \begin{pmatrix} u_i \\ p_i \end{pmatrix}.$$

We found that performance was essentially in the asymptotic range for $h = 1/32$, and all results are for this mesh size.

We present three types of data: iteration counts, estimates for convergence factors, and plots of residual norms as functions of operation counts. The iteration counts are averages over three runs of the number of steps needed to satisfy the stopping criterion; these are shown in Table 4. The estimates for asymptotic convergence factors are the averages of $(\|\bar{R}_{5+i}\|_2/\|\bar{R}_5\|_2)^{1/i}$ over all steps after step five; here $\bar{R}_k$ represents the average of the k'th residual norm over the three runs. These are shown in Table 5. We chose step five rather than step zero because performance was often better in the first few steps than later, when the asymptotic behavior is seen. Finally, Figures 2 – 5 plot the averages of the residual norms against operation counts.

We make the following observations on these results.

1. Where it is applicable, multigrid requires the smallest number of iterations and has the smallest convergence factors. MG/ILU is superior to MG/DGS in these measures. These observations agree with those of [35]. In addition, where it is applicable, MG/ILU requires the smallest number of operations. However, multigrid is only effective for discretizations where velocities and pressures are defined on the same grid.

FIG. 2. *Operation counts for finite difference discretization.*

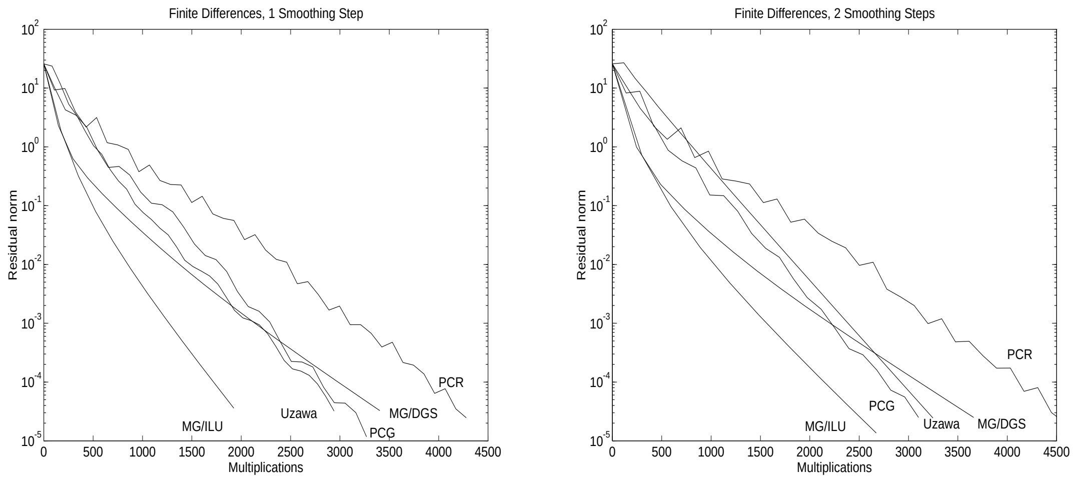


FIG. 3. *Operation counts for* $P_1(h)P_0(2h)$ *finite element discretization.*

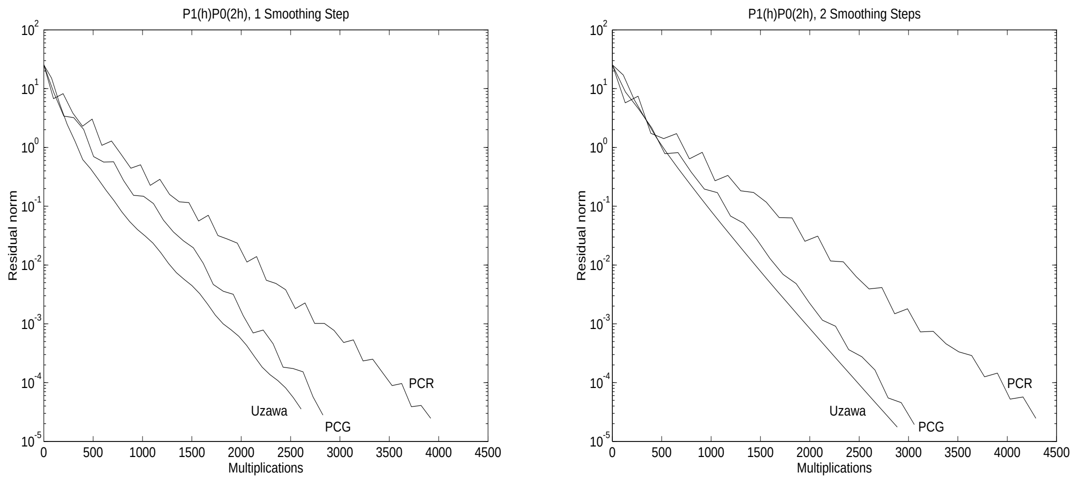


FIG. 4. *Operation counts for $P_1(h)P_1(2h)$ finite element discretization.*

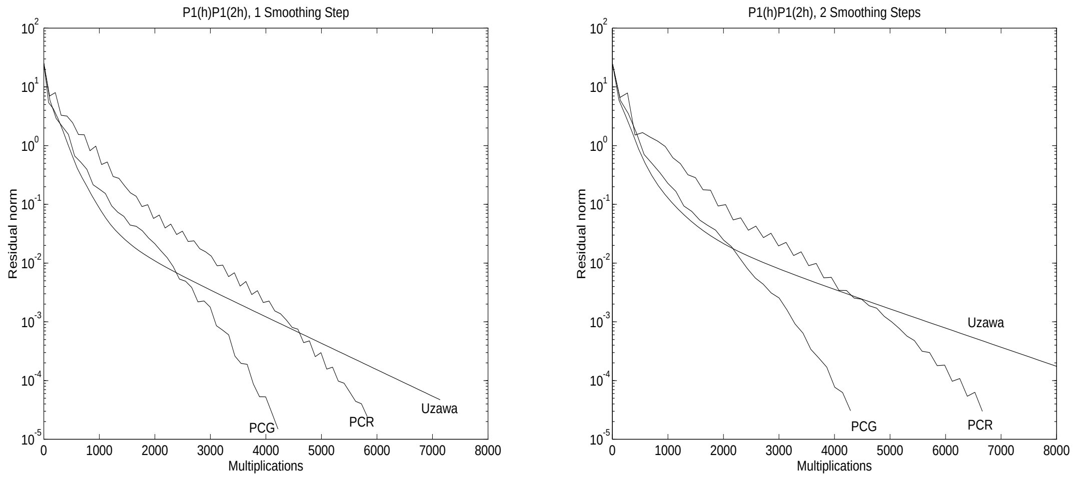
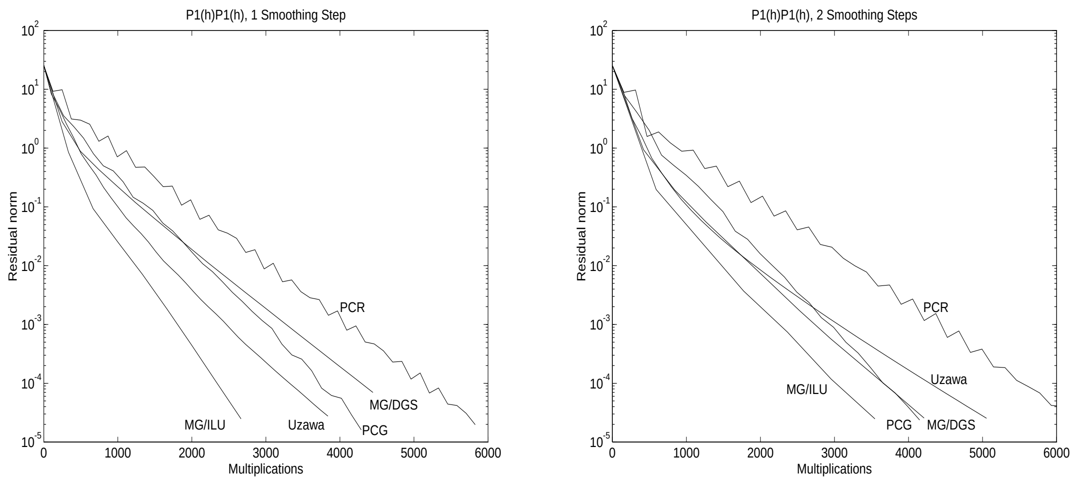


FIG. 5. *Operation counts for $P_1(h)P_1(h)$ finite element discretization.*



2. The Krylov subspace methods and MG/DGS are roughly equal in cost. The Krylov subspace methods are more widely applicable than multigrid.
3. The performances of all these methods are very close. In terms of operation counts, the ratio of costs of the most expensive and least expensive method is no worse than 2.3.
4. No Krylov subspace method is clearly superior to the others. PCG exhibits a somewhat faster convergence rate than PCR, and the Uzawa algorithm is surprisingly competitive with the other two methods. This appears to derive from the dependence of PCG and PCR on both the spectral condition number κ from (13) and the accuracy of the preconditioner Q_A as an approximation to A; for both these methods, the iteration counts go down in all cases when the number of smoothing steps in Q_A increases. The Uzawa method appears to be less sensitive to the accuracy of Q_A. The values of κ for the three problems are:

Finite differences	4.14	$P_1(h)P_1(2h)$	22.71
$P_1(h)P_0(2h)$	4.87	$P_1(h)P_1(h)$	9.91

The Uzawa method is least effective for the $P_1(h)P_1(2h)$ discretization, which has the largest condition number.
5. The Uzawa and PCG methods depend on choices of iteration parameters. These can be estimated relatively inexpensively (e.g., using a coarse grid for the Uzawa method, and a few steps of the power method for PCG), but this increases the cost of these methods and makes implementing them considerably more difficult. In contrast, PCR is independent of parameters except for those needed for the multigrid preconditioning, and it is therefore easier to implement. Thus, there is a tradeoff between these methodologies: PCR converges slightly more slowly than PCG and, often, than the Uzawa method, but it has a simpler implementation.
6. For each of the solution strategies except PCG, it is less expensive to use one smoothing step than two.

Acknowledgements. The author wishes to thank David Silvester for a careful reading of a preliminary version of this paper, and Andy Wathen for some helpful remarks.

REFERENCES

[1] D. Arnold, F. Brezzi, and M. Fortin, *A stable finite element for the Stokes equations*, Calcolo, 21 (1984), pp. 337–344.

[2] K. Arrow, L. Hurwicz, and H. Uzawa, *Studies in Nonlinear Programming*, Stanford University Press, Stanford, CA, 1958.

[3] S. F. Ashby, T. A. Manteuffel, and P. E. Saylor, *A taxonomy for conjugate gradient methods*, SIAM J. Numer. Anal., 27 (1990), pp. 1542–1568.

[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, *A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems*, Math. Comp., 50 (1988), pp. 1–17.

[6] A. Brandt and N. Dinar, *Multigrid solutions to elliptic flow problems*, in Numerical Methods for Partial Differential Equations, S. V. Parter, ed., Academic Press, New York, 1979, pp. 53–147.

[7] F. Brezzi and M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer-Verlag, New York, 1991.

[8] F. Brezzi and J. Pitkäranta, *On the stabilisation of finite element approximations of the Stokes problem*, in Efficient Solutions of Elliptic Systems, W. Hackbusch, ed., Braunschweig, Vieweg, 1984, pp. 11–19. Notes on Numerical Fluid Mechanics, Vol 10.

[9] R. Chandra, S. C. Eisenstat, and M. H. Schultz, *The modified conjugate residual method for partial differential equations*, in Advances in Computer Methods for Partial Differential Equations II, R. Vichnevetski, ed., IMACS, New Brunswick, 1977, pp. 13–19.

[10] N. Decker, *Note on the parallel efficiency of the Frederickson-McBryan multigrid algorithm*, SIAM J. Sci. Stat. Comput., 12 (1991), pp. 208–220.

[11] H. C. Elman and G. H. Golub, *Inexact and preconditioned Uzawa algorithms for saddle point problems*, Tech. Report UMIACS-TR-93-41, Institute for Advanced Computer Studies, University of Maryland, 1993. To appear in SIAM J. Numer. Anal.

[12] M. Fortin and R. Glowinski, *Augmented Lagrangian Methods: Applications to the Numerical Solution of Boundary-Value Problems*, North-Holland, New York, 1983.

[13] P. O. Frederickson and O. A. McBryan, *Normalized convergence rates for the PSMG method*, SIAM J. Sci. Stat. Comput., 12 (1991), pp. 221–229.

[14] R. Glowinski, *Numerical Methods for Nonlinear Variational Problems*, Springer-Verlag, New York, 1984.

[15] G. H. Golub and C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, second ed., 1989.

[16] P. M. Gresho and R. L. Sani, *On pressure boundary conditions for the incompressible Navier-Stokes equations*, Int. J. Numer. Meth. Fluids, 7 (1987), pp. 1111–1145.

[17] M. Gunzburger, *Finite Element Methods for Viscous Incompressible Flows*, Academic Press, San Diego, 1989.

[18] W. Hackbusch, *Multi-Grid Methods and Applications*, Springer-Verlag, Berlin, 1985.

[19] F. H. Harlow and J. E. Welch, *Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface*, The Physics of Fluids, 8 (1965), pp. 2182–2189.

[20] S. F. McCormick, ed., *Multigrid Methods*, SIAM, Philadelphia, 1987.

[21] R. A. Nicolaides, *Analysis and convergence of the MAC scheme I*, SIAM J. Numer. Anal., 29 (1992), pp. 1579–1591.

[22] J. Pitkäranta and T. Saarinen, *A multigrid version of a simple finite element method for the Stokes problem*, Math. Comp., 45 (1985), pp. 1–14.

[23] A. Ramage and A. J. Wathen, *Iterative Solution Techniques for the Navier-Stokes Equations*, Tech. Report 93-01, School of Mathematics , University of Bristol, 1993. To appear in Int. J. Numer. Meth. Fluids.

[24] T. Rusten and R. Winther, *A preconditioned iterative method for saddle point problems*, SIAM J. Matr. Anal. Appl., 13 (1992), pp. 887–904.

[25] D. Silvester, *Optimal low order finite element methods for incompressible flow*, Comp. Meths. Appl. Mech. Engrg., 111 (1994), pp. 357–368.

[26] D. Silvester and A. Wathen, *Fast Iterative Solution of Stabilized Stokes Systems. Part II: Using Block Preconditioners*, Tech. Report 218, Mathematics Dept., University of Manchester Institute of Science and Technology, 1992. To appear in SIAM J. Numer. Anal., 1994.

[27] D. B. Szyld and O. B. Widlund, *Variational analysis of some conjugate gradient methods*, East-West J. of Numer. Math., 1 (1991), pp. 1–25.

[28] S. P. Vanka, *Block-implicit multigrid solution of Navier-Stokes in primitive variables*, J. Comput. Phys., 65 (1986), pp. 138–158.

[29] R. Verfürth, *A combined conjugate gradient-multigrid algorithm for the numerical solution of the Stokes problem*, IMA J. Numer. Anal., 4 (1984), pp. 441–455.

[30] ———, *A multilevel algorithm for mixed problems*, SIAM J. Numer. Anal., 21 (1984), pp. 264–271.

[31] A. Wathen and D. Silvester, *Fast iterative solution of stabilized Stokes systems. Part I: Using simple diagonal preconditioners*, SIAM J. Numer. Anal., 30 (1993), pp. 630–649.

[32] A. J. Wathen, *Realistic eigenvalue bounds for the Galerkin mass matrix*, IMA J. Numer. Anal., 7 (1987), pp. 449–457.

[33] B. D. Welfert, *Convergence of Inexact Uzawa Algorithms for Saddle Point Problems*, tech. report, Mathematics Department, University of Arizona, 1993.

[34] P. Wesseling, *An Introduction to Multigrid Methods*, John Wiley & Sons, New York, 1992.

[35] G. Wittum, *Multi-grid methods for the Stokes and Navier-Stokes equations*, Numer. Math., 54 (1989), pp. 543–564.

[36] ———, *On the convergence of multi-grid methods with transforming smoothers*, Numer. Math., 57 (1990), pp. 15–38.