SIAM J. SCI. COMPUT.
Vol. 39, No. 5, pp. S461–S476
© 2017 Society for Industrial and Applied Mathematics

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

ACCELERATING THE UZAWA ALGORITHM*

NGUYENHO HO†, SARAH D. OLSON‡, AND HOMER F. WALKER‡

Abstract. The Uzawa algorithm is an iterative method for the solution of saddle-point problems, which arise in many applications, including fluid dynamics. Viewing the Uzawa algorithm as a fixed-point iteration, we explore the use of Anderson acceleration (also known as Anderson mixing) to improve the convergence. We compare the performance of the preconditioned Uzawa algorithm with and without acceleration on several steady Stokes and Oseen problems for incompressible flows. For perspective, we include in our comparison GMRES with two different preconditioners. The results indicate that the accelerated preconditioned Uzawa algorithm converges significantly faster than the algorithm without acceleration and is competitive with the other methods considered.

Key words. Uzawa algorithm, saddle-point problems, preconditioning, Anderson acceleration, Stokes problems, Oseen problems, incompressible flows

AMS subject classifications. 65F10, 76D07, 65H10

DOI. 10.1137/16M1076770

1. Introduction.

A saddle-point problem is a block 2×2 linear system of the form

$$\underbrace{\begin{pmatrix} A & B^T \\ B & C \end{pmatrix}}_{\mathcal{A}} \begin{pmatrix} u \\ p \end{pmatrix} = \underbrace{\begin{pmatrix} f \\ g \end{pmatrix}}_{b}. \tag{1.1}$$

Here, A is assumed to be invertible but may be symmetric or nonsymmetric. We make no assumptions about the rank of B but note that rank deficiency in B can usually be remedied without difficulty in practice [4, sect. 4.2]. Applications in which saddle-point problems arise include computational fluid dynamics [21, 29], constrained optimization [33], linear elasticity, economics, and many other areas [4].

Many methods for solving saddle-point problems have been developed and analyzed, including direct and iterative methods (see [4] for an extensive review). Here, we consider the *Uzawa algorithm* [2], a well-known and easily implemented iterative method. Extensions of the classical Uzawa algorithm include the augmented Lagrangian formulation [6, 27] and the inexact Uzawa algorithm (also known as the Arrow–Hurwicz method) [7, 11, 15, 14, 22, 25, 30, 34]. A relaxation parameter appears in the algorithm, and methods for determining optimal values of this are outlined in [3]. Additionally, the use of preconditioners to increase the speed of convergence of these methods has been studied by a number of authors [7, 8, 11, 12, 15, 14, 23].

In practice, the iterates produced by the Uzawa algorithm often converge undesirably slowly. In this paper, we view the algorithm as a fixed-point iteration and consider the use of *Anderson acceleration* [1], also known as *Anderson mixing*, to improve

*Received by the editors May 24, 2016; accepted for publication (in revised form) February 24, 2017; published electronically October 26, 2017.
http://www.siam.org/journals/sisc/39-5/M107677.html
Funding: This work was supported in part by U.S. National Science Foundation grants DMS 1455270 and DMS 1337943, and U.S. Department of Energy award DE-SC0004880.
†Corresponding author. Department of Mathematical Sciences, University of Cincinnati, Cincinnati, OH 45221 (hono@uc.edu).
‡Department of Mathematical Sciences, Worcester Polytechnic Institute, Worcester, MA 01609-2280 (sdolson@wpi.edu, walker@wpi.edu).

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

the convergence. This acceleration method (denoted henceforth by AA) has enjoyed considerable success in accelerating fixed-point iterations in electronic-structure computations (see [19] and the references therein), simulations of transport phenomena [10, 24], and coupled fluid-solid interface problems [20]. It has also been effective in accelerating Picard iterations in variably saturated flow and advection-diffusion simulations [28, 26]. To the best of our knowledge, it has not been previously used with the Uzawa algorithm or other iterations for saddle-point problems.

It is unusual to apply AA in the context of a linear problem such as (1.1), for which there are already many solution methods. Indeed, as we note below, AA coupled with the Uzawa algorithm is very closely related to GMRES [31] with a certain preconditioning applied to (1.1). Our goal is to present AA as an easily implemented and economical way to mitigate the often slow convergence of Uzawa iterates and to demonstrate performance of the accelerated algorithm on several model problems.

In the following, we focus on applying AA to the Uzawa algorithm for the Stokes and Oseen problems of steady incompressible flow. The use of AA with any iterative method in this fluids setting appears to be new. In this context, $C = 0$ in (1.1) when applying a stable finite-element discretization. (The extension to nonzero C is straightforward.) We briefly consider the classical standard form of the Uzawa algorithm but focus primarily on the preconditioned form, since the standard form without preconditioning converges too slowly to be practical on the problems of interest. In section 2, we outline the standard and preconditioned algorithms. In section 3, we first note a particular way in which the Uzawa algorithm can be recast as a fixed-point iteration; we then describe AA and discuss its relationship with GMRES in this case. In section 4, we report on numerical experiments, in which we compare the performance of the Uzawa algorithm, with and without acceleration, in several Stokes and Oseen flow scenarios. For perspective, we include the closely related preconditioned GMRES method and also, in one test, GMRES preconditioned with the relaxed dimensional factorization (RDF) preconditioner developed in [5]. In section 5, we offer a summary discussion and conclusions.

2. Variations of the Uzawa algorithm.

2.1. Standard Uzawa.

The standard Uzawa algorithm consists of a coupled iteration for the variables p and u. Given an initial guess for p_0, the iteration is

$$u_{k+1} = A^{-1}(f - B^T p_k), \tag{2.1a}$$

$$p_{k+1} = p_k + \omega(Bu_{k+1} - g), \tag{2.1b}$$

where $\omega > 0$ is a relaxation parameter. It has been shown previously that if one eliminates u_{k+1} from (2.1b) by using (2.1a), then the Uzawa method is equivalent to a Richardson iteration, which can be used to determine an optimal ω in terms of the maximum and minimum eigenvalues of the Schur complement $\mathcal{S} = BA^{-1}B^T$, where A is symmetric positive-definite [4]. Several variations of this standard algorithm have been developed, including preconditioned versions that we describe next.

2.2. Preconditioned Uzawa.

We consider the preconditioned Uzawa algorithm in the general form given in [7]:

$$u_{k+1} = u_k + Q_A^{-1}\left(f - (Au_k + B^T p_k)\right), \tag{2.2a}$$

$$p_{k+1} = p_k + \omega Q_B^{-1}(Bu_{k+1} - g), \tag{2.2b}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

where Q_A and Q_B are preconditioners.[1] When $Q_A = A$, this becomes the preconditioned Uzawa algorithm considered in [15]; when $Q_B = I$ as well, it reduces to the standard Uzawa algorithm in (2.1a)–(2.1b). Note also that (2.2a)–(2.2b) require an initial guess for u_0 as well as p_0 if $Q_A \neq A$.

Often, A is symmetric positive-definite (SPD). In this case, suitable preconditioners Q_A and Q_B should be SPD as well, and various choices have appeared in the literature. For example, [15] includes an instance of (2.2a)–(2.2b) with $Q_A = A$ and Q_B a scaled diagonal or tridiagonal matrix derived from the mass matrix of a finite-element discretization. Other examples appear in [7], with Q_A a multigrid V-cycle and $Q_B = I$, and in [11], with Q_A an incomplete-Cholesky factorization of A and Q_B a scaled identity matrix. Choices of Q_A and Q_B when A is nonsymmetric are similarly varied. For example, in both [8] and [12], the symmetric part $A_S = \frac{1}{2}(A + A^T)$ is assumed to be SPD. Choices of Q_A and Q_B in [8] include taking Q_A to be a multigrid V-cycle on A_S and Q_B to be a scaled identity matrix; choices in [12] include $Q_A = A$ and $Q_B = B(A_S)^{-1}B^T$.

3. Accelerating preconditioned Uzawa. Although the developments in this section are for the preconditioned Uzawa algorithm (2.2a)–(2.2b), it is straightforward to adapt them to the standard algorithm (2.1a)–(2.1b) by taking $Q_A = A$ and $Q_B = I$.

3.1. Preconditioned Uzawa as a fixed-point iteration. We begin by rewriting the preconditioned Uzawa algorithm as a fixed-point iteration. From (2.2a)–(2.2b),

$$\underbrace{\begin{pmatrix} Q_A & 0 \\ B & -\frac{1}{\omega}Q_B \end{pmatrix}}_{\mathcal{M}} \begin{pmatrix} u_{k+1} \\ p_{k+1} \end{pmatrix} = \underbrace{\begin{pmatrix} Q_A - A & -B^T \\ 0 & -\frac{1}{\omega}Q_B \end{pmatrix}}_{\mathcal{N}} \begin{pmatrix} u_k \\ p_k \end{pmatrix} + \begin{pmatrix} f \\ g \end{pmatrix}, \tag{3.1}$$

which immediately gives the following fixed-point version of preconditioned Uzawa:

$$\begin{pmatrix} u_{k+1} \\ p_{k+1} \end{pmatrix} = G \begin{pmatrix} u_k \\ p_k \end{pmatrix} \equiv \mathcal{M}^{-1} \left(\mathcal{N} \begin{pmatrix} u_k \\ p_k \end{pmatrix} + \begin{pmatrix} f \\ g \end{pmatrix} \right). \tag{3.2}$$

The explicit form of the right-hand side of (3.2) is

$$\begin{pmatrix} Q_A^{-1}(Q_A - A) & -Q_A^{-1}B^T \\ \omega Q_B^{-1}BQ_A^{-1}(Q_A - A) & I - \omega Q_B^{-1}BQ_A^{-1}B^T \end{pmatrix} \begin{pmatrix} u_k \\ p_k \end{pmatrix} + \begin{pmatrix} Q_A^{-1}f \\ \omega Q_B^{-1}(BQ_A^{-1}f - g) \end{pmatrix},$$

where $BQ_A^{-1}B^T \approx BA^{-1}B^T$, the Schur complement. If $Q_A = A$, then (3.2) becomes

$$\begin{pmatrix} u_{k+1} \\ p_{k+1} \end{pmatrix} = \begin{pmatrix} 0 & -A^{-1}B^T \\ 0 & I - \omega Q_B^{-1}BA^{-1}B^T \end{pmatrix} \begin{pmatrix} u_k \\ p_k \end{pmatrix} + \begin{pmatrix} A^{-1}f \\ \omega Q_B^{-1}(BA^{-1}f - g) \end{pmatrix}. \tag{3.3}$$

Formally, the iteration (3.3), like (3.2), requires an initial guess for u_0 as well as p_0. However, in (3.3), this can be arbitrary.

3.2. Anderson acceleration. The outline of AA given below applies to a general fixed-point iteration $\xi_{i+1} = G(\xi_k)$ that begins with some initial value ξ_o. In the context of interest, $\xi = (u,\ p)^T$ and G is defined in (3.2). In the algorithm, m denotes

[1] This iteration is called an *inexact* Uzawa algorithm in [7, 8], since the ultimate interest there is iteratively solving the linear subproblems. It is not the same as the inexact Uzawa algorithm of [15].

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

the maximum number of stored residuals, which is necessarily finite in practice.

Anderson Acceleration

Given ξ_0 and $m \geq 1$, set $\xi_1 = G(\xi_0)$.

For $k = 1, 2, \ldots$

Set $m_k = \min\{m, k\}$.

Set $F_k = (\tilde{f}_{k-m_k}, \ldots, \tilde{f}_k)$, where $\tilde{f}_i = G(\xi_i) - \xi_i$.

Determine $\alpha^{(k)} = \left(\alpha_0^{(k)}, \ldots, \alpha_{m_k}^{(k)}\right)^T$ that solves

$$\min_{\alpha=(\alpha_0,\ldots,\alpha_{m_k})^T} \|F_k\alpha\|_2 \qquad \text{s.t.} \qquad \sum_{i=0}^{m_k} \alpha = 1.$$

Set $\xi_{k+1} = \sum_{i=0}^{m_k} \alpha_i^{(k)} G(\xi_{k-m_k+i})$.

This basic form of the algorithm suffices for our purposes. In practice, it can be modified in various ways, e.g., by incorporating step-relaxation or modifying m_k to maintain acceptable conditioning of the least-squares problem (see [32]).

The rationale underlying AA is that, on a linear fixed-point problem such as the problem of interest here, one has $\sum_{i=0}^{m_k} \alpha_i^{(k)} G(\xi_{k-m_k+i}) = G(\sum_{i=0}^{m_k} \alpha_i^{(k)} \xi_{k-m_k+i})$. It follows that, in this case, ξ_{k+1} is obtained by applying G to the point within the affine subspace containing $\xi_k, \ldots, \xi_{k-m_k}$ that has a minimal fixed-point residual.

In practice, the constrained least-squares problem in $m_k + 1$ variables is usually reformulated as an equivalent unconstrained problem in m_k variables. See [19, 32] for a particular reformulation that can be implemented efficiently and, in our experience, is numerically sound. In any case, as the algorithm proceeds, both the number of stored residual vectors and the amount of arithmetic per iteration increase up to maxima determined by m. It is important to note, though, that only a single G-evaluation is required at each iteration, as in the underlying fixed-point iteration. Thus the cost of applying AA to a fixed-point iteration is only some additional storage and arithmetic, discussed further in section 4.2.

3.3. Matrix-splitting preconditioners. As noted in the introduction, the Uzawa algorithm accelerated with AA is closely related to GMRES applied to (1.1) with a certain preconditioning. To describe this relationship in more detail, we note that the matrix $\mathcal{A}$ in (1.1) with $C = 0$ can be split as follows:

$$\mathcal{A} = \mathcal{M} - \mathcal{N} = \begin{pmatrix} Q_A & 0 \\ B & -\frac{1}{\omega}Q_B \end{pmatrix} - \begin{pmatrix} Q_A - A & -B^T \\ 0 & -\frac{1}{\omega}Q_B \end{pmatrix}, \tag{3.4}$$

where $\mathcal{M}$ and $\mathcal{N}$ are defined in (3.1) for Q_A and Q_B in (2.2a)–(2.2b).[2]

The fixed-point form of preconditioned Uzawa (3.2) is a stationary iteration determined by this splitting. Suppose that AA is applied to this stationary iteration without truncation, i.e., with $m_k = k$ at each iteration, so that all previous residuals are used in the least-squares problems. Suppose also that unrestarted GMRES is applied to the left-preconditioned system

$$\mathcal{M}^{-1}\mathcal{A}\begin{pmatrix} u \\ p \end{pmatrix} = \mathcal{M}^{-1}\begin{pmatrix} f \\ g \end{pmatrix}, \tag{3.5}$$

[2]When $Q_A = A$, this splitting has been considered by others; see section 8.1 of [4] and the references therein.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

starting from the same initial approximate solution. Denote the kth iterates of the two algorithms by ξ_k^{AA} and ξ_k^{GMRES}, respectively. Then, provided GMRES does not stagnate before the solution is found, Corollary 2.10 in [32] asserts that the two algorithms are "essentially equivalent" in the sense that, until the solution is found, $\xi_k^{\mathrm{GMRES}} = \sum_{i=0}^{m_k} \alpha_i^{(k)} \xi_i^{\mathrm{AA}}$ and $\xi_{k+1}^{\mathrm{AA}} = G(\xi_k^{\mathrm{GMRES}})$, where G is given by (3.2).

This "essential equivalence" seldom strictly holds in practice, since it is almost always necessary to truncate AA and to restart GMRES. We explore in section 4 how well this "essential equivalence" is borne out in the problems of interest there. For now, we note that

$$\mathcal{M}^{-1} = \begin{pmatrix} Q_A^{-1} & 0 \\ \omega Q_B^{-1} B Q_A^{-1} & -\omega Q_B^{-1} \end{pmatrix},$$

and so the main requirement of using this preconditioner in GMRES is one solve with each of Q_A and Q_B per iteration. Additionally, (3.5) corresponds to

$$\begin{pmatrix} Q_A^{-1}A & Q_A^{-1}B^T \\ \omega Q_B^{-1} B Q_A^{-1}(A - Q_A) & \omega Q_B^{-1} B Q_A^{-1} B^T \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} Q_A^{-1} f \\ \omega Q_B^{-1}(B {Q_A}^{-1} f - g) \end{pmatrix}.$$

When $Q_A = A$, this simplifies to

$$\begin{pmatrix} I & A^{-1}B^T \\ 0 & \omega Q_B^{-1} B A^{-1} B^T \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} A^{-1} f \\ \omega Q_B^{-1}(BA^{-1} f - g) \end{pmatrix}. \tag{3.6}$$

We refer to GMRES applied to (3.6) as PGMRES in what follows. When $Q_A = A$, it follows from the discussion above that untruncated AA applied to preconditioned Uzawa is "essentially equivalent" in the above-defined sense to unrestarted PGMRES. When $Q_B = I$ as well, preconditioned Uzawa reduces to standard Uzawa, and it follows that untruncated AA applied to standard Uzawa is "essentially equivalent" to unrestarted GMRES applied to (3.6) with $Q_B = I$, i.e.,

$$\begin{pmatrix} I & A^{-1}B^T \\ 0 & \omega B A^{-1} B^T \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} A^{-1} f \\ \omega(BA^{-1} f - g) \end{pmatrix}, \tag{3.7}$$

which mainly requires one solve with A at each iteration.

The forms of (3.6) and (3.7) are especially convenient for applying GMRES. For example, the matrix-vector products required for GMRES applied to (3.6) mainly require only one solve with each of A and Q_B, in addition to matrix-vector products with B and B^T. Alternatively, one may prefer first to solve the preconditioned or unpreconditioned Schur-complement system, i.e.,

$$Q_B^{-1} B A^{-1} B^T p = Q_B^{-1}(BA^{-1} f - g) \quad \text{or} \quad BA^{-1}B^T p = BA^{-1} f - g,$$

and then to form $u = A^{-1}(f - B^T p)$. This approach has the advantage of working with shorter vectors in the Arnoldi process within GMRES; however, it requires an additional solve with A at the end in order to recover u.[3] In any case, the preconditioned Schur-complement system points to the need for Q_B to be a good preconditioner for $BA^{-1}B^T$. See sections 5 and 10 of [4] for more discussion of the Schur-complement approach and associated preconditioning considerations.

[3] In the important special case in which A is SPD, one would very likely use the preconditioned conjugate-gradient (PCG) method to solve the Schur-complement system. Since the approximate solution for p is updated at each PCG iteration, one can also update the approximate solution for u concurrently at very little cost. In this case, the Schur-complement approach is likely to be preferred.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

4. Numerical experiments. In this section, we show experimental results comparing the performance of the methods discussed above on the Stokes and Oseen problems in several steady incompressible-flow scenarios in two space dimensions.

4.1. Fluid equations. We consider the steady-state incompressible Navier–Stokes equations

$$-\nu\Delta u + (u\cdot\nabla)u + \nabla p = f, \tag{4.1}$$

$$\nabla\cdot u = 0 \tag{4.2}$$

on a bounded, connected domain Ω in $\mathbb{R}^2$. In (4.1)–(4.2), ν is the kinematic viscosity, $u = (u_x, u_y)^T : \Omega \to \mathbb{R}^2$ is the fluid velocity, $p : \Omega \to \mathbb{R}$ is the pressure field, and $f : \Omega \to \mathbb{R}^2$ is a force field. The boundary conditions on $\partial\Omega$ are expressed as $\mathcal{B}u = g$, where $\mathcal{B}$ is a boundary operator.

Our interest is in the steady Stokes and Oseen equations, which are obtained from (4.1) in two different ways. These are given, respectively, by

$$-\nu\Delta u + \nabla p = f, \tag{4.3}$$

$$-\nu\Delta u + (v\cdot\nabla)u + \nabla p = f, \tag{4.4}$$

together with the incompressibility condition (4.2) and boundary conditions $\mathcal{B}u = g$. In the Stokes equations, the nonlinear term $(u\cdot\nabla)u$ in (4.1) is dropped with the assumption that viscous terms dominate and the convection term can be neglected. In the Oseen equations, the term $(u\cdot\nabla)u$ in (4.1) is linearized and replaced by $(v\cdot\nabla)u$, where v is known. For example, v may be a previous approximate solution.

Finite-element discretizations of the Oseen and Stokes equations lead to saddle-point systems of the form (1.1). We focus on the stable Q2−Q1 finite-element discretization with $C = 0$, which satisfies the Ladyžhenskaya–Babuška–Brezzi inf-sup condition [4]. This is a sufficient condition for (1.1) to have a unique solution and ensures that the condition number of the Schur complement $BA^{-1}B^T$ is bounded and independent of the mesh size [4]. The matrix A is SPD for the Stokes equation and is nonsymmetric for the Oseen equation. In our experiments, we used the IFISS MATLAB package [17] with the Q2−Q1 finite-element discretization to obtain the necessary matrices and vectors for the discretized problems. In experiments with the Oseen equations, we took v to be the fifth Picard iterate produced by IFISS applied to the Navier–Stokes equations (4.1)–(4.2). The initial approximate solution for Picard iteration is the solution to the Stokes equation. In our numerical studies, the initial conditions for the pressure and the velocity are set to zero.

4.2. Methods and preconditioners. The methods we compare and our notation for them are as follows:

NASU	nonaccelerated standard Uzawa (2.1a)–(2.1b)
ASU	standard Uzawa accelerated with AA
NAPU	nonaccelerated preconditioned Uzawa (2.2a)–(2.2b)
APU	preconditioned Uzawa accelerated with AA
PGMRES	GMRES applied to the preconditioned system (3.6)

With the accelerated algorithms, we indicate the maximum number of stored residuals in AA (i.e., the truncation parameter m) in parentheses; e.g., ASU(20) denotes ASU with $m = 20$ in AA. Similarly, we show restart parameters in PGMRES, e.g., PGMRES(20). In each experiment, truncation and restart parameters were chosen to be the same.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

We discuss in broad terms the storage and computational costs of the algorithms of major interest: AA, preconditioned Uzawa, and PGMRES. In counting arithmetic operations, we assume $m \ll n$ and include only $\mathcal{O}(n)$ operations. The implementation of AA used in our tests is briefly described in [32, sect. 4]. For each iteration after the mth, this requires about $(6m-3)n$ arithmetic operations and storage of roughly $2(m+1)$ n-vectors. No matrix operations involving A, B, Q_A, or Q_B are required directly by AA; these are required only by the underlying Uzawa iteration. From (3.3) and (3.6), one sees that preconditioned Uzawa and PGMRES require the same matrix multiplications and solves at each iteration. In applications, these are likely to dominate the computational costs of both algorithms. In our tests, we used the standard modified Gram–Schmidt implementation of PGMRES(m). Over a cycle of m steps, this requires about $[m(m+3)+1]n$ arithmetic operations and storage of roughly $(m+1)$ n-vectors. For comparison, note that over a set of m steps after the mth, AA requires about $m(6m-3)n$ arithmetic operations. If the costs of matrix multiplications and solves required by the algorithms are not dominant, then the arithmetic disadvantage of AA may be a consideration, as may its storage disadvantage when n is very large.

We consider the Uzawa algorithm without preconditioning only in the first test scenario in section 4.3. Thereafter, we consider only the preconditioned algorithm and fix the preconditioners in (2.2a)–(2.2b) as follows:

- Stokes equations: $Q_A = A$ and Q_B is the tridiagonal part of the pressure mass matrix (cf. [15]).
- Oseen equations: $Q_A = A$ and

$$Q_B^{-1} = \left(BM_1^{-1}B^T\right)^{-1} BM_1^{-1}AM_1^{-1}B^T \left(BM_1^{-1}B^T\right)^{-1}, \tag{4.5}$$

  where M_1 is the diagonal of the velocity mass matrix (cf. [16]).

The preconditioner in (4.5) is developed by Elman et al. in [16] as an approximation of the inverse of the Schur complement. It is a *least-squares commutator* preconditioner specifically referred to as the *scaled BFBt method* in [16]; see also [13] and [18, sect. 9.2.3]. The matrices in (4.5) are easy to obtain since they are available from the saddle-point system. Applying this preconditioner involves two Poisson-type solves and one velocity solve.

Note that PGMRES involves the preconditioner Q_B, since it appears in (3.6). In experiments with preconditioned Uzawa, Q_B in (3.6) is as described above. In experiments with standard (unpreconditioned) Uzawa, $Q_B = I$, and PGMRES is just GMRES applied to (3.7).

In all numerical experiments, the stopping criterion was

$$\frac{||r_k||_2}{||b||_2} \leq 10^{-6}, \tag{4.6}$$

where r_k is the kth residual and b is the right-hand side vector in (1.1). In the following, we report on the number of iterations required by the algorithms of interest to satisfy (4.6) in the test cases described. Since MATLAB was used for the experiments, these are perhaps more meaningful than timing data.

4.3. Channel flow. The first test problem is channel flow on the square domain $\Omega = [-1,1]\times[-1,1]$. In this, a parabolic inflow $u = (1-y^2, 0)^T$ is prescribed on the left side of the channel at $x = -1$. The top and bottom boundary conditions at $y = \pm 1$ are set to a no-flow Dirichlet condition $u = (0,0)^T$, and a Dirichlet condition is used

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

for the outflow at $x = 1$. With these boundary conditions, we consider the Stokes equations (4.3)–(4.2) with viscosity $\nu = 1$, for which the standard Uzawa algorithm is effective but slow. In our experiments, these equations were discretized on five uniform grids: 16×16, 32×32, 64×64, 128×128, and 256×256. These correspond to 659, 2467, 9539, 37507, and 148739 unknowns, respectively.

Although our main interest is in the preconditioned Uzawa algorithm, for completeness we first consider the standard (unpreconditioned) Uzawa algorithm (2.1a)–(2.1b) applied to the Stokes equations. Specifically, we compare the numbers of iterations necessary to satisfy (4.6) for ASU, NASU, and PGMRES. (We remind the reader that PGMRES in this standard Uzawa case is GMRES applied to (3.7).) Our preliminary experiments showed that, on this problem, the performance of ASU is not sensitive to the truncation parameter m in the AA algorithm. In the experiments reported here, we somewhat arbitrarily chose $m = 20$ and took this to be the restart parameter in PGMRES as well. Also, following [15], we took the relaxation parameter in (2.1b) to be $\omega = 2/(\lambda_m + \lambda_M)$, where λ_m and λ_M are the minimum and maximum eigenvalues of the Schur complement $BA^{-1}B^T$, which is SPD since A is SPD.

The results of our experiments are shown in Table 1. One sees that ASU(20) consistently required far fewer iterations than NASU to satisfy (4.6); thus acceleration was very effective in these tests. As expected from the discussion in section 3.3, ASU(20) is roughly (but not exactly) "essentially equivalent" to PGMRES(20). Additionally, the convergence of all methods appears to be mesh-independent (i.e., does not slow appreciably) as the grid is refined. This property is noted for NASU on the Stokes problem in [4, sect. 8.1]. To illustrate the convergence history of the methods, we show in Figure 1 (A) the log residual-norm plots for ASU(20), NASU, and PGMRES(20) on the 64×64 grid.

TABLE 1
Iterations required to satisfy (4.6) *for the channel-flow Stokes problem with* $\nu = 1$.

Grid	ASU (20)	NASU	PGMRES (20)
16×16	20	261	19
32×32	26	268	29
64×64	26	228	29
128×128	25	175	26
256×256	22	119	25

We now consider the preconditioned Uzawa algorithm (2.2a)–(2.2b) applied to this Stokes problem. The preconditioners Q_A and Q_B are as described in subsection 4.2, and PGMRES is GMRES applied to (3.6). The methods of main interest are APU, NAPU, and PGMRES. As in the unpreconditioned case, convergence of APU is not sensitive to the truncation parameter m in AA. Since convergence is somewhat faster than in the unpreconditioned case, we used $m = 10$ in the experiments reported here. We also observed that convergence is not strongly dependent on the relaxation parameter ω in (2.2b) in this case, and we used $\omega = 1$ in these experiments.

Table 2 shows the number of iterations required by the methods to satisfy (4.6). As in the unpreconditioned case, acceleration significantly reduced the necessary number of iterations, and again the convergence of all methods appears to be mesh-independent. Figure 1 (B) shows the log residual-norm plots for the four methods corresponding to the 32×32 grid case in Table 2. The log residual-norm plots for APU and PGMRES decrease rapidly. In contrast, the curve for NAPU decreases much more slowly.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

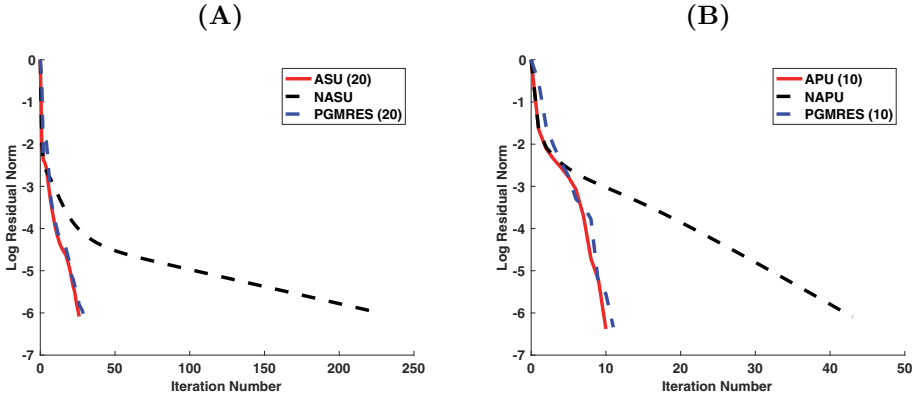

FIG. 1. *Log residual-norm plots for the channel-flow Stokes problem with* (A) *the standard Uzawa algorithm* (2.1a)–(2.1b) *for* $\nu = 1$ *on the* 64×64 *grid and* (B) *the preconditioned Uzawa algorithm* (2.2a)–(2.2b) *for* $\nu = 1$ *on the* 32×32 *grid.*

TABLE 2
Iterations required to satisfy (4.6) *for the channel-flow Stokes problem with* $\nu = 1$.

Grid	APU (10)	NAPU	PGMRES (10)
16×16	10	44	10
32×32	10	43	11
64×64	11	41	12
128×128	11	38	12
256×256	11	36	12

4.4. Lid-driven cavity. The second test problem is the "leaky" lid-driven cavity problem on $\Omega = [-1, 1] \times [-1, 1]$. With the "lid" moving from left to right, the top boundary condition is $u = (1, 0)^T$ at $y = 1$, $-1 \le x \le 1$. No-flow boundary conditions $u = (0, 0)^T$ are imposed on the other three sides.[4] In our experiments, we again discretized the problem on the five uniform grids used in section 4.3, resulting in the same respective numbers of unknowns. In these and all subsequent experiments below, results were obtained using only the preconditioned Uzawa algorithm (2.2a)–(2.2b).

We first consider the incompressible Stokes equations (4.3)–(4.2) and take $\nu = 1$ in (4.3). In this case, it was sufficient to use $\omega = 1$ in (2.2b), and $m = 10$ led to adequately fast convergence of APU. The results for this case are summarized in Table 3. As in the channel-flow problem in section 4.3, APU(10) and PGMRES(10) performed about the same at all grid sizes in our tests, and all three required significantly fewer iterations to converge than NAPU. Again, the convergence of all methods appears to be mesh-independent.

We next consider the Oseen equations (4.4)–(4.2) and report results for $\nu = 0.1$, 0.01, and 0.001. For this more challenging problem, we used $m = 20$ in APU in all tests. Also, for $\nu = 0.1$ and $\nu = 0.01$, we experimentally determined values of ω in

[4] In this "enclosed flow" case, B had rank deficiency one, and the coefficient matrix in the preconditioned system (3.6) was singular. We verified numerically that in both the Stokes and Oseen problems, (3.6) was consistent and the null space and range of the coefficient matrix intersected only at zero. Consequently, it follows from [9, Thm. 2.6] that GMRES should compute the solution of (3.6) without breaking down. Also, in applying Q_B^{-1} in (4.5) in the Oseen case, solving with $(BM_1^{-1}B^T)$ involved only right-hand sides in the range of B, and so solutions were well-defined in the pseudo-inverse sense. We observed no problematic behavior with either PGMRES or the preconditioned Uzawa iteration (3.3) in our experiments on this test problem.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

TABLE 3
Iterations required to satisfy (4.6) *for the leaky lid-driven cavity Stokes problem with* $\nu = 1$.

Grid	APU(10)	NAPU	PGMRES(10)
16×16	12	49	12
32×32	12	50	14
64×64	12	50	14
128×128	11	49	14
256×256	11	48	14

(2.2b) that approximately minimized the numbers of NAPU iterations. For $\nu = 0.001$, we were unable to find an ω for which NAPU converged, and so we determined values of ω that approximately minimized the numbers of APU iterations (except for the 16×16 grid, for which no method converged for any value of ω). The results are summarized in Table 4. In the table, "> 1000" indicates that the iterates appeared to be converging but failed to satisfy (4.6) within 1000 iterations; an asterisk indicates that the iterates did not appear to be converging.

TABLE 4
Iterations required to satisfy (4.6) *for the leaky lid-driven cavity Oseen problem. The values of* ω *in* (2.2b) *shown were experimentally determined to approximately minimize the numbers of* NAPU *iterations when* $\nu = 0.1$ *and* $\nu = 0.01$ *and to approximately minimize the numbers of* APU *iterations when* $\nu = 0.001$.

	Grid	ω	APU(20)	NAPU	PGMRES(20)
$\nu = 0.1$	16×16	0.64	10	11	10
	32×32	0.45	12	17	12
	64×64	0.29	15	27	15
	128×128	0.16	18	46	18
	256×256	0.087	28	77	41
$\nu = 0.01$	16×16	1.2	16	51	16
	32×32	0.74	21	91	20
	64×64	0.43	23	148	24
	128×128	0.24	31	244	40
	256×256	0.12	32	402	48
$\nu = 0.001$	16×16	*	*	*	*
	32×32	1.6	99	*	378
	64×64	0.87	111	*	600
	128×128	0.31	99	*	> 1000
	256×256	0.17	113	*	> 1000

One sees from Table 4 that, for $\nu = 0.1$ and $\nu = 0.01$, acceleration not only significantly reduced iteration numbers but also mitigated mesh dependence, which is at most mild for APU(20) but pronounced for NAPU. Moreover, for $\nu = 0.001$, acceleration greatly improved robustness, with APU(20) succeeding except on the 16×16 grid, while NAPU failed on all grids.

A striking aspect of the results in Table 4 is the difference in performance of APU(20) and PGMRES(20) when the number of iterations is greater than 20. As discussed in section 3.3, the "essential equivalence" of APU and PGMRES is theoretically assured only when the acceleration in APU is not truncated and PGMRES is not restarted. Evidently, the performance differences seen in Table 4 result from truncating APU and restarting PGMRES, with the latter having more significant negative effects on performance than the former, perhaps because all previous information about the iteration is discarded at each PGMRES restart.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

To further investigate the effects of the choice of the restart value on PGMRES convergence on this problem, we tried a number of different restart values with $\nu = 0.001$ on the 64×64 grid. The results are in Table 5. One sees that performance comparable to that of APU(20) in Table 4 was achieved, but only with a restart value much greater than 20 and a correspondingly greater cost of storage and arithmetic.

TABLE 5
Iterations required by restarted PGMRES *to satisfy* (4.6) *for the leaky lid-driven cavity Oseen problem with* $\nu = 0.001$ *on the* 64×64 *grid with* $\omega = 0.87$ *(cf. Table* 4*).*

Restart value	20	30	40	50	60	70
Iterations	600	343	190	167	131	98

We also explored the sensitivity of APU convergence to the parameter m, the number of stored residuals in AA. In Figure 2, we illustrate how the convergence of APU varies with m for the Stokes and Oseen equations. For the Stokes equations, the log residual-norm plots (shown in Figure 2 (A)) decrease smoothly and show relatively little sensitivity to m, with little meaningful difference in the plots for the four values of m. For the Oseen equations, the plots (shown in Figure 2 (B)) decrease smoothly for the most part, although there is a modest plateau region in the middle iterations. There is also somewhat greater sensitivity to m in this more challenging case.

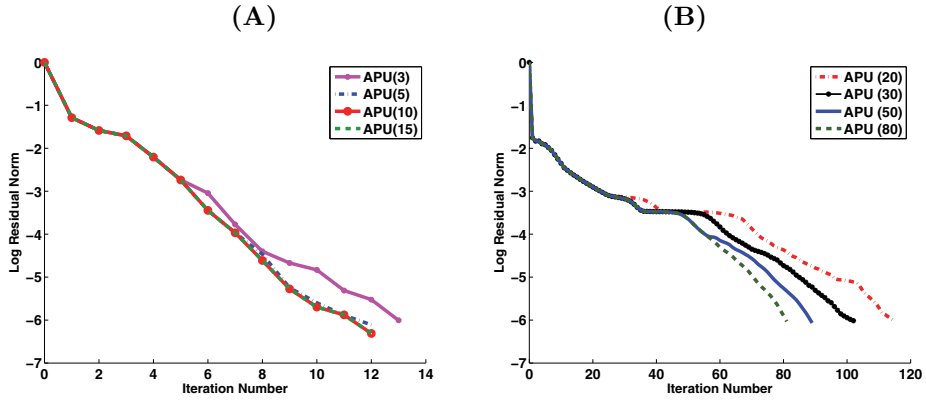

FIG. 2. *Log residual norm plots for the leaky lid-driven cavity problem using different numbers of stored residuals* m *in* APU *on the* 64×64 *grid:* (A) *the Stokes equations with* $\nu = 1$ *in* (4.3) *and* $\omega = 1$ *in* (2.2b) *(cf. Table* 3*);* (B) *the Oseen equations with* $\nu = 0.001$ *in* (4.3) *and* $\omega = 0.87$ *in* (2.2b) *(cf. Table* 4*).*

4.5. Flow past an obstacle. The next test problem corresponds to channel flow inside the domain $\Omega = [0, 8] \times [-1, 1]$ with a square object located at $[0, 2] \times [0.5, 0.5]$. The boundary conditions are as in subsection 4.3, except Neumann boundary conditions are set for the outflow. We consider four different grid sizes: 16×32, 32×64, 64×128, and 128×256 corresponding to 2488, 9512, 37168, and 146912 unknowns, respectively. We again focus on the preconditioned Uzawa algorithm (2.2a)–(2.2b) with and without acceleration applied to the Oseen equations (4.4)–(4.2) and take $m = 20$ in APU(m).

For additional perspectives that reflect recent developments in splitting preconditioners for incompressible-flow saddle-point problems, we consider GMRES preconditioned with the relaxed dimensional factorization (RDF) preconditioner developed

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

in [5], which is shown in [5] to be effective on the problems of interest here. With $C = 0$, we follow [5] and write (1.1) equivalently as

$$\begin{pmatrix} A & B^T \\ -B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ -g \end{pmatrix}. \tag{4.7}$$

For two-dimensional flow, we partition A and B as

$$A = \begin{pmatrix} A_1 & 0 \\ 0 & A_2 \end{pmatrix}, \quad B = (B_1, B_2).$$

The RDF preconditioner is

$$M_\beta = \begin{pmatrix} A_1 & -\frac{1}{\beta} B_1^T B_2 & B_1^T \\ 0 & A_2 & B_2^T \\ -B_1 & -B_2 & \beta I \end{pmatrix}, \tag{4.8}$$

where β is a problem-dependent parameter. A factorization of M_β that allows efficient solves is described in [5].

We refer to GMRES preconditioned with M_β simply as RDF. In the numerical results reported in Table 6, we used values of β that were determined through auxiliary experiments to approximately minimize the number of RDF iterations in each case.

TABLE 6

Iterations required to satisfy (4.6) *for the obstacle-flow Oseen problem. The values of* ω *in* (2.2b) *shown were experimentally determined to approximately minimize the numbers of* NAPU *iterations for each value of* ν. *In the* RDF *column, the numbers in parentheses are the values of* β *used in each case.*

	Grid	ω	APU(20)	NAPU	PGMRES(20)	RDF(20)
$\nu = 0.1$	16×32	0.72	12	13	11	16 (0.044)
	32×64	0.49	13	17	13	17 (0.014)
	64×128	0.34	16	25	15	17 (0.005)
	128×256	0.21	19	39	18	16 (0.001)
$\nu = 0.01$	16×32	1.61	34	138	36	18 (0.096)
	32×64	1.12	25	67	26	20 (0.041)
	64×128	0.85	16	21	16	22 (0.013)
	128×256	0.49	14	19	16	24 (0.004)
$\nu = 0.005$	16×32	1.28	55	489	66	21 (0.100)
	32×64	0.76	50	332	57	19 (0.050)
	64×128	0.76	28	77	30	31 (0.013)
	128×256	0.52	16	36	16	34 (0.005)

The results in Table 6 show that, as in the previous test problems, acceleration was effective in reducing the number of preconditioned Uzawa iterations. When $\nu = 0.1$, acceleration also mitigated mesh dependence somewhat, although it is fairly mild for NAPU in this case. Interestingly, when $\nu = 0.01$ and $\nu = 0.005$, the iteration numbers required by APU(20), NAPU, and PGMRES(20) all decrease as the grid is refined. For all ν values, the iteration numbers for APU(20) and PGMRES(20) consistently differ by little, in contrast to the previous problem. The iteration numbers for RDF(20) suggest at most mild mesh dependence for that method. They are about the same as those of APU(20) and PGMRES(20) when $\nu = 0.1$. For the two smaller values of ν, they are somewhat smaller for the two coarser grids and somewhat larger for the two finer grids.

To further illustrate the convergence of the methods, we show in Figure 3 (A) the log residual-norm plots of APU(20), NAPU, PGMRES(20), and RDF(20) for the case

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

$\nu = 0.01$ and the 32×64 grid. Also, we note that the choice of m in AA does matter for this Oseen problem. In Figure 3 (B), we show results for different choices of m for this problem with $\nu = 0.005$ on the 32×64 grid. The figure clearly indicates that APU(m) converges faster as m increases. It is also notable that significant improvement over NAPU was obtained with m as small as five.

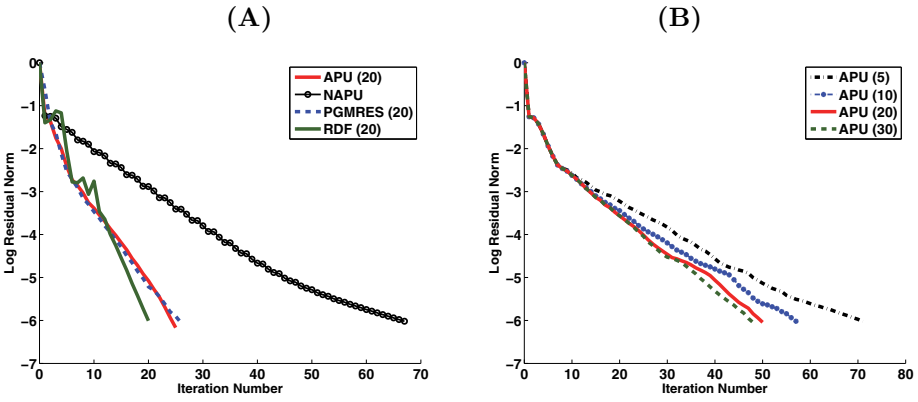

FIG. 3. *Log residual-norm plots for the obstacle-flow Oseen problem on the* 32×64 *grid:* (A) *all methods with* $\nu = 0.01$*;* (B) APU(m) *for different* m *with* $\nu = 0.005$ *(cf. Table* 6*).*

4.6. Backward-facing step. Our final test problem is flow over a backward-facing step. In this, the domain is $\Omega = [-1, 0] \times [0, 1] \cup [0, 5] \times [-1, 1]$, which has a downward "step" at $x = 0$. As in subsection 4.3, a parabolic flow profile is imposed at the inflow boundary ($x = -1$ and $0 \le y \le 1$). Neumann boundary conditions are imposed at the outflow boundary ($x = 5$ and $-1 < y < 1$). At the other walls, a no-flow boundary is prescribed. We study the problem using four different grids having the general form $2^n \times 3 \cdot 2^n$ for $n = 4, 5, 6, 7$, corresponding to 1747, 6659, 25987, and 102659 unknowns, respectively. As in subsection 4.5, we consider the Oseen equations (4.4)–(4.2) with $\nu = 0.1$, 0.01, and 0.005 and report on the performance of the methods with $m = 20$ in APU(m). The results are summarized in Table 7.

TABLE 7

Iterations required to satisfy (4.6) *for the backward-facing step Oseen problem. The values of* ω *in* (2.2b) *shown in the table were experimentally determined to approximately minimize the numbers of* NAPU *iterations for each value of* ν*.*

	Grid	ω	APU(20)	NAPU	PGMRES(20)
$\nu = 0.1$	16×48	0.64	11	12	11
	32×96	0.45	14	17	14
	64×192	0.29	17	26	17
	128×384	0.17	20	40	19
$\nu = 0.01$	16×48	0.93	24	61	24
	32×96	0.55	15	25	14
	64×192	0.46	14	24	14
	128×384	0.38	19	32	20
$\nu = 0.005$	16×48	0.44	41	400	40
	32×96	0.42	28	112	31
	64×192	0.32	20	49	20
	128×384	0.31	21	36	20

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

The results in Table 7 are qualitatively very similar to those in Table 6. As in that case, acceleration was effective in reducing the number of preconditioned Uzawa iterations and also in mitigating the rather mild mesh dependence of NAPU when $\nu = 0.1$. When $\nu = 0.01$ and $\nu = 0.005$, the table shows some tendency for the iteration numbers of APU(20), NAPU, and PGMRES(20) to actually decrease as the grid is refined, although it is not as pronounced or consistent as in the previous case. The iteration numbers for APU(20) and PGMRES(20) are very similar and show no significant mesh dependence. We conclude this section by showing in Figure 4 the performance of all methods in the case of $\nu = 0.005$ with the 32×96 grid.

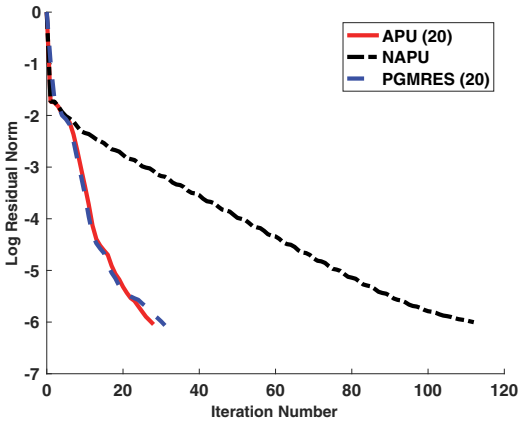

FIG. 4. *Log residual-norm plots for all methods on the backward-facing step Oseen problem with viscosity* $\nu = 0.005$ *and the* 32×96 *grid (cf. Table* 7*).*

5. Discussion and conclusions. We have introduced *Anderson acceleration* (AA) as an easily implemented and economical way of improving the convergence of the Uzawa algorithm for solving a saddle-point system (1.1). While such systems arise in many settings, the applications of primary interest here are the steady Stokes and Oseen problems of incompressible flow.

We show that, viewed as a fixed-point iteration, the preconditioned Uzawa algorithm is a stationary iteration for (1.1) determined by the splitting (3.4). It follows from results in [32] that preconditioned Uzawa accelerated with untruncated AA is "essentially equivalent" in a certain sense to unrestarted GMRES applied to the preconditioned system (3.5). When the Uzawa preconditioner Q_A in (2.2a) is equal to A, the preconditioned system (3.5) simplifies to (3.6), a form especially convenient for applying GMRES, referred to as PGMRES in this case. Similar results hold for standard Uzawa as a special case, with the simplified form of the preconditioned system given by (3.7). Since the "essential equivalence" does not strictly hold when PGMRES is restarted and AA is truncated, we included restarted PGMRES in our tests to compare its performance with that of Uzawa accelerated with truncated AA.

In all test cases, acceleration reduced the number of iterations required by the Uzawa algorithm to satisfy the convergence criterion (4.6). The reduction was usually significant, and in many cases it was dramatic. Accelerated Uzawa usually required about the same numbers of iterations or modestly fewer than restarted PGMRES; however, in one case, that of the leaky lid-driven cavity problem (see Table 4), the accelerated method APU(20) was significantly more robust than PGMRES(20) and

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

often required significantly fewer iterations when the iterates from both methods converged. The results in Table 5 suggest that the PGMRES iterates can be made to converge in as few iterations as those of APU(20) in this test case, but only by taking the restart value to be much larger than 20. In the tests involving Stokes flow (see Tables 1–3), all methods showed little adverse mesh dependence. This is not surprising, in view of the mesh-independence property of nonaccelerated Uzawa for Stokes flow noted in [4, sect. 8.1].

In the case of flow past an obstacle, we also included results for GMRES using the relaxed dimensional factorization (RDF) preconditioner developed in [5]. For this problem, accelerated Uzawa usually required about the same numbers of iterations as RDF, although not in every test case (see Table 6). For the leaky lid-driven cavity and backward facing step test cases, we also observed that accelerated Uzawa performs similarly to RDF; however, those results are not reported here since they are similar to results in [5]. We view the accelerated Uzawa as a method competitive with RDF in applications to Stokes and Oseen problems.

REFERENCES

[1] D. G. Anderson, *Iterative procedures for nonlinear integral equations*, J. Assoc. Comput. Mach., 12 (1965), pp. 547–560.

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

[3] C. Bacuta, B. McCracken, and L. Shu, *Residual reduction algorithms for nonsymmetric saddle point problems*, J. Comput. Appl. Math., 235 (2011), pp. 1614–1628.

[4] M. Benzi, G. H. Golub, and J. Liesen, *Numerical solution of saddle point problems*, Acta Numerica, 2005 (2005), pp. 1–137.

[5] M. Benzi, M. Ng, Q. Niu, and Z. Wang, *A relaxed dimensional factorization preconditioner for the incompressible Navier–Stokes equations*, J. Comput. Phys., 230 (2011), pp. 6185–6202.

[6] M. Benzi, M. A. Olshanskii, and Z. Wang, *Modified augmented Lagrangian preconditioners for the incompressible Navier–Stokes equations*, Internat. J. Numer. Methods Fluids, 66 (2010), pp. 486–508.

[7] J. H. Bramble, J. E. Pasciak, and A. T. Vassilev, *Analysis of the inexact Uzawa algorithm for saddle point problems*, SIAM J. Numer. Anal., 34 (1997), pp. 1072–1092, https://doi.org/10.1137/S0036142994273343.

[8] J. H. Bramble, J. E. Pasciak, and A. T. Vassilev, *Uzawa type algorithms for nonsymmetric saddle point problems*, Math. Comp., 69 (1999), pp. 667–689.

[9] P. N. Brown and H. F. Walker, *GMRES on (nearly) singular systems*, SIAM J. Matrix Anal. Appl., 18 (1997), pp. 37–51, https://doi.org/10.1137/S0895479894262339.

[10] M. T. Calef, E. D. Fichtl, J. S. Warsa, M. Berndt, and N. N. Carlson, *Nonlinear Krylov acceleration applied to a discrete ordinates formulation of the k-eigenvalue problem*, J. Comput. Phys., 238 (2013), pp. 188–209.

[11] Z. H. Cao, *Fast Uzawa algorithm for generalized saddle point problems*, Appl. Numer. Math., 46 (2003), pp. 157–171.

[12] M. R. Cui, *Analysis of iterative algorithms of Uzawa type for saddle point problems*, Appl. Numer. Math., 50 (2004), pp. 133–146.

[13] H. Elman, V. E. Howle, J. Shadid, D. Silvester, and R. Tuminaro, *Least squares preconditioners for stabilized discretizations of the Navier–Stokes equations*, SIAM J. Sci. Comput., 30 (2007), pp. 290–311, https://doi.org/10.1137/060655742.

[14] H. C. Elman, *Multigrid and Krylov subspace methods for the discrete Stokes equations*, Internat. J. Numer. Methods Fluids, 22 (1996), pp. 755–770.

[15] H. C. Elman and G. H. Golub, *Inexact and preconditioned Uzawa algorithms for saddle point problems*, SIAM J. Numer. Anal., 31 (1994), pp. 1645–1661, https://doi.org/10.1137/0731085.

[16] H. C. Elman, V. E. Howle, J. Shadid, R. Shuttleworth, and R. Tuminaro, *Block preconditioners based on approximate commutators*, SIAM J. Sci. Comput., 27 (2006), pp. 1651–1668, https://doi.org/10.1137/040608817.

[17] H. C. Elman, A. Ramage, and D. J. Silvester, *IFISS: A Matlab toolbox for modelling*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

incompressible flow, ACM Trans. Math. Software, 33 (2007), 14.

[18] H. C. Elman, D. J. Silvester, and A. J. Wathen, *Finite Elements and Fast Iterative Solvers: With Applications in Incompressible Fluid Dynamics*, Oxford University Press, Oxford, 2014.

[19] H. Fang and Y. Saad, *Two classes of multisecant methods for nonlinear acceleration*, Numer. Linear Algebra Appl., 16 (2009), pp. 197–221.

[20] V. Ganine, N. J. Hills, and B. L. Lapworth, *Nonlinear acceleration of coupled fluid-structure transient thermal problems by Anderson mixing*, Internat. J. Numer. Methods Fluids, 71 (2013), pp. 939–959.

[21] R. Glowinski, *Numerical Methods for Nonlinear Variational Problems*, Springer Series in Computational Physics, Springer, New York, 1984.

[22] Q. Hu and J. Zou, *Nonlinear inexact Uzawa algorithms for linear and nonlinear saddle-point problems*, SIAM J. Optim., 16 (2006), pp. 798–825, https://doi.org/10.1137/S1052623403428683.

[23] A. Klawonn, *An optimal preconditioner for a class of saddle point problems with a penalty term*, SIAM J. Sci. Comput., 19 (1998), pp. 540–552, https://doi.org/10.1137/S1064827595279575.

[24] D. Kuzmin, *Linearity-preserving flux correction and convergence acceleration for constrained Galerkin schemes*, J. Comput. Appl. Math., 236 (2012), pp. 2317–2337.

[25] Y. Lin and Y. Cao, *A new nonlinear Uzawa algorithm for generalized saddle point problems*, Appl. Math. Comput., 175 (2006), pp. 1432–1454.

[26] K. Lipnikov, D. Svyatskiy, and Y. Vassilevski, *Anderson acceleration for nonlinear finite volume scheme for advection-diffusion problems*, SIAM J. Sci. Comput., 35 (2013), pp. A1120–A1136, https://doi.org/10.1137/120867846.

[27] W. Liu and S. Xu, *An new improved Uzawa method for finite element solution of Stokes problem*, Comput. Mech., 27 (2001), pp. 305–310.

[28] P. A. Lott, H. F. Walker, C. S. Woodward, and U. M. Yang, *An accelerated Picard method for nonlinear systems related to variably saturated flow*, Adv. Water Resources, 38 (2012), pp. 92–101.

[29] A. Quarteroni and A. Valli, *Numerical Approximation of Partial Differential Equations*, Springer-Verlag, Berlin, Heidelberg, 1994.

[30] W. Queck, *The convergence factor of preconditioned algorithms of the Arrow–Hurwicz type*, SIAM J. Numer. Anal., 26 (1989), pp. 1016–1030, https://doi.org/10.1137/0726057.

[31] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869, https://doi.org/10.1137/0907058.

[32] H. F. Walker and P. Ni, *Anderson acceleration for fixed-point iterations*, SIAM J. Numer. Anal., 49 (2011), pp. 1715–1735, https://doi.org/10.1137/10078356X.

[33] M. H. Wright, *Interior methods for constrained optimization*, Acta Numerica, 1 (1992), pp. 341–407.

[34] W. Zulehner, *Analysis of iterative methods for saddle point problems: A unified approach*, Math. Comp., 71 (2001), pp. 479–505.

Downloaded 11/06/17 to 152.3.102.254. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.