REDUCING NONLINEAR COMPLEXITY OF TWO-LEVEL NONLINEAR ADDITIVE SCHWARZ PRECONDITIONED INEXACT NEWTON METHOD*

FENG-NAN HWANG† AND XIAO-CHUAN CAI‡

Abstract. We propose and test a new two-level nonlinear additive Schwarz preconditioned inexact Newton algorithm(ASPIN). The two-level nonlinear preconditioner combines a local nonlinear additive Schwarz preconditioner and a global *linear* coarse preconditioner. This approach is more attractive than the two-level method introduced in [6, 14] which is nonlinear on both levels. Since the coarse mesh function evaluation only requires to solve a linear coarse system rather than a nonlinear coarse system derived from the discretization of original partial differential equations, the overall computational cost can be reduced considerately. Our parallel numerical results based on a driven cavity incompressible flow problem show that the new two-level ASPIN is nearly scalable with respect to the number of processors, the fine mesh size, and Reynolds number provided that the coarse mesh size is fine enough.

Key words. Nonlinear Schwarz preconditioning, finite element, parallel processing, Newton method, incompressible Navier-Stokes equations

AMS subject classifications. 65F10, 65H10, 65H20, 65N06, 65N12, 65N22, 65N55, 65Y05, 76D05

1. Introduction.

We focus on the parallel numerical solution of large, sparse nonlinear systems of equations arising from the finite element or finite difference discretization of nonlinear partial differential equations. Such systems appear in many computational science and engineering applications, such as the simulation of fluid flows [4, 18]. Several classes of techniques are available, for example [3, 7, 15], including Newton type methods, multigrid type methods, nonlinear Krylov subspace type methods, and continuation type methods. However, for some difficult problems, such as incompressible Navier-Stokes equations with high Reynolds number (Re), none of the methods works well, except the continuation method, which is often too slow to be considered practical. In this paper, we develop a general nonlinear preconditioning technique that is fast, robust and scalable for solving nonlinear systems of equations. Our nonlinearly preconditioned method is based on the inexact Newton method with backtracking technique(INB). Briefly speaking, let

$$F(x^*) = 0 \tag{1.1}$$

be a nonlinear system of equations and $x^{(0)}$ a given initial guess. Assume $x^{(k)}$ is the current approximate solution. Then a new approximate solution $x^{(k+1)}$ of (1.1) can be computed by first find an inexact Newton direction $s^{(k)}$ satisfying

$$||F(x^{(k)}) - F'(x^{(k)})s^{(k)}||_2 \leq \eta_k ||F(x^{(k)})||_2,$$

then obtain $x^{(k+1)}$ with $x^{(k+1)} = x^{(k)} - \lambda^{(k)} s^{(k)}$, where the parameter $\lambda^{(k)}$ is computed via backtracking. In INB, the scalar η_k is often called the "forcing term", which determines how accurately the Jacobian system needs to be solved by some iterative methods, such as a Krylov subspace type method, GMRES [17]. If the forcing terms are chosen small enough, the algorithm reduces to the exact Newton algorithm. The scalar $\lambda^{(k)}$ is selected so that

$$f(x^{(k)} - \lambda^{(k)} s^{(k)}) \leq f(x^{(k)}) - \alpha \lambda^{(k)} \nabla f(x^{(k)})^T s^{(k)}, \tag{1.2}$$

*The work was partially supported by the Department of Energy, DE-FC02-01ER25479, and by the National Science Foundation, CCR-0219190, ACI-0072089 and ACI-0305666.

†Department of Applied Mathematics, University of Colorado at Boulder, Boulder, CO 80309 (hwangf@colorado.edu)

‡Department of Computer Science, University of Colorado at Boulder, Boulder, CO 80309 (cai@cs.colorado.edu)

where the merit function f is defined as $||F(x)||_2^2/2$. Here, a line search technique [7] is employed to determine the step length $\lambda^{(k)}$ and the parameter α is used to assure that the reduction of f is sufficient.

Although INB has the desirable property of local fast convergence, like other nonlinear iterative methods, INB is very fragile. It converges rapidly for a well-selected set of parameters (for example, certain initial guess, certain range of Re), but diverges if we slightly change some of the parameters. On the other hand, it may converge well at the beginning of the iterations, then suddenly stall for no apparent reason. In [5, 6, 11, 12] some nonlinear preconditioning methods were developed, and the Newton convergence becomes not sensitive to these unfriendly parameters if INB is applied to a nonlinearly preconditioned system

$$\mathcal{F}(x^*) = 0 \tag{1.3}$$

instead. Here the word "preconditioner" refers to the fact that systems (1.1) and (1.3) have the same solution and the new system (1.3) is better conditioned, both linearly and nonlinearly. The preconditioner is constructed using a nonlinear additive Schwarz method. To a certain extend, the robustness problem is solved since the methods converge for all Reynolds numbers and all mesh sizes, but the parallel scalability remains an issue. To improve the processor scalability, a two-level method was then proposed in [6], which works well if the number of processors is not large. For large number of processors, the *nonlinear* coarse solver takes too much CPU and communication times. In this paper, we suggest a combined *linear and nonlinear* additive Schwarz preconditioner and show that using a linear coarse solver we can maintain the nonlinear robustness and reduce the nonlinear complexity considerably.

The paper is organized as follows. In Section 2, we discuss a two-level nonlinear additive Schwarz preconditioned inexact Newton algorithm and introduce a new linear coarse solver, which plays the central role in the scalability of the algorithm. We also describe two-dimensional steady-state incompressible Navier-Stokes equations and their finite element discretization, which is taken as an example to illustrate the applicability of the method. Then, in Section 3, we present some numerical results obtained on a parallel computer for a lid-driven cavity flow problem. Particularly, we focus on the parallel linear and nonlinear scalability of the method. Finally, Section 4 presents some concluding remarks.

2. Nonlinear preconditioning algorithms.

In this section, we describe a two-level nonlinear preconditioner based on a combination of local nonlinear additive Schwarz preconditioners and a global linear coarse preconditioner. The local nonlinear preconditioners make the method more robust in the sense that the method converges for a wide range of Reynolds numbers and mesh sizes, and the linear coarse preconditioner makes the method more scalable in the sense that the number of iterations does not depend much on the number of parallel processors. This is very important for solving large scale problems on massively parallel computers.

2.1. A model problem.

Although our method should work for general nonlinear systems of equations, we restrict our discussion on a two-component system(velocity and pressure) resulting from the finite element discretization of incompressible Navier-Stokes equations. The generalization to other multi-component problems is straightforward. Consider steady-state incompressible Navier-Stokes equations in the primitive variable form [10, 16]:

$$\left\{ \begin{array}{lll} \boldsymbol{u} \cdot \nabla \boldsymbol{u} - 2\nu \nabla \cdot \epsilon(\boldsymbol{u}) + \nabla p = 0 & \text{in} & \Omega, \\ \nabla \cdot \boldsymbol{u} = 0 & \text{in} & \Omega, \\ \boldsymbol{u} = \boldsymbol{g} & \text{on} & \Gamma, \end{array} \right. \tag{2.1}$$

where $\boldsymbol{u}$ is the velocity, p is the pressure, $\nu = 1/Re$ is the dynamic viscosity, and $\epsilon(\boldsymbol{u}) = 1/2(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T)$ is the symmetric part of the velocity gradient. The pressure p is determined up to a constant. To make p unique, we impose an additional condition $\int_\Omega p\, dx = 0$. To discretize

(2.1), we use a stabilized $Q_1 - Q_1$ finite element method ([2, 8]). For simplicity, we consider only a uniform square mesh $\mathcal{T}^h = \{K\}$. Let V^h and P^h be a pair of finite element spaces for the velocity and pressure, given by

$$\begin{aligned} V^h &= \{\boldsymbol{v} \in (C^0(\Omega) \cap H^1(\Omega))^2 : \ \boldsymbol{v}|_K \in Q_1(K)^2, \ K \in \mathcal{T}_h \ \} \\ P^h &= \{p \in C^0(\Omega) \cap L^2(\Omega) : \ p|_K \in Q_1(K), \ K \in \mathcal{T}_h\}. \end{aligned}$$

The weighting and trial velocity function spaces V_0^h and V_g^h are

$$V_0^h = \{\boldsymbol{v} \in V^h : \boldsymbol{v} = 0 \text{ on } \Gamma\} \text{ and } V_g^h = \{\boldsymbol{v} \in V^h : \ \boldsymbol{v} = g \text{ on } \Gamma\}.$$

Similarly, let the finite element space P_0^h be both the weighting and trial pressure function spaces:

$$P_0^h = \left\{ p \in P^h : \int_\Omega p\, dx = 0 \right\}.$$

Following [8], the stabilized finite element method for steady-state incompressible Navier-Stokes equations reads: Find $\boldsymbol{u}^h \in V_g^h$ and $p^h \in P_0^h$, such that

$$B(\boldsymbol{u}^h, p^h; \boldsymbol{v}, q) = 0 \qquad \forall (\boldsymbol{v}, q) \in V_0^h \times P_0^h \tag{2.2}$$

with

$$\begin{aligned} &B(\boldsymbol{u}, p; \boldsymbol{v}, q) = ((\nabla \boldsymbol{u}) \cdot \boldsymbol{u}, \boldsymbol{v}) + (2\nu\epsilon(\boldsymbol{u}), \epsilon(\boldsymbol{v})) - (\nabla \cdot \boldsymbol{v}, p) - (\nabla \cdot \boldsymbol{u}, q) + \\ &\sum_{K \in \mathcal{T}_h} ((\nabla \boldsymbol{u}) \cdot \boldsymbol{u} + \nabla p, \tau((\nabla \boldsymbol{v}) \cdot \boldsymbol{v} - \nabla q))_K + (\nabla \cdot \boldsymbol{u}, \delta \nabla \cdot \boldsymbol{v}) \end{aligned}$$

We use the stability parameters δ and τ as suggested in [8]. The stabilized finite element formulation (2.2) can be written as a nonlinear algebraic system

$$F(x) = 0, \tag{2.3}$$

which is often large, sparse, and highly nonlinear when the value of Reynolds number is large. The vector x corresponds to the nodal values of $\boldsymbol{u}^h = (u_1^h, u_2^h)$ and p^h in (2.2).

2.2. Subdomain partition, and one-level nonlinear preconditioner. To define parallel Schwarz type preconditioners [19] we need to partition the finite element mesh $\mathcal{T}^h$ introduced in the previous section. Let $\{\Omega_i^h, i = 1, ..., N\}$ be a non-overlapping subdomain partition whose union covers the entire domain Ω and its mesh $\mathcal{T}^h$. We use $\mathcal{T}_i^h$ to denote the collection of mesh points in Ω_i^h. To obtain overlapping subdomains, we expand each subdomain Ω_i^h to a larger subdomain $\Omega_i^{h,\delta}$ with the boundary $\partial\Omega_i^{h,\delta}$. Here δ is an integer indicating the degree of overlap. We assume that $\partial\Omega_i^{h,\delta}$ does not cut any elements of $\mathcal{T}^h$. Similarly, we use $\mathcal{T}_i^{h,\delta}$ to denote the collection of mesh points in $\Omega_i^{h,\delta}$.

Now, we define the subdomain velocity space as

$$V_i^h = \{v^h \in V^h \cap \left(H^1(\Omega_i^{h,\delta})\right)^2 : v^h = 0 \text{ on } \partial\Omega_i^{h,\delta}\}$$

and the subdomain pressure space as

$$P_i^h = \{p^h \in P^h \cap L^2(\Omega_i^{h,\delta}) : p^h = 0 \text{ on } \partial\Omega_i^{h,\delta} \backslash \Gamma\},$$

On the physical boundaries, we impose Dirichlet conditions according to the original equations (2.1). On the artificial boundaries, we assume both $\boldsymbol{u} = 0$ and $p = 0$. Similar boundary conditions were used in [13].

Let $R_i : V^h \times P^h \to V_i^h \times P_i^h$ be a restriction operator, which returns all degrees of freedom (both velocity and pressure) associated with the subspace $V_i^h \times P_i^h$. R_i is an $3n_i \times 3n$ matrix with values of either 0 or 1, where n and n_i are the total number of mesh points in $\mathcal{T}^h$ and $\mathcal{T}_i^{h,\delta}$, respectively, and $\sum_{i=1}^N 3n_i \geq 3n$. Note that for $Q_1 - Q_1$ elements, we have three variables per mesh point, two for the velocity and one for the pressure. Then, the interpolation operator R_i^T can be defined as the transpose of R_i. The multiplication of R_i (and R_i^T) with a vector does not involve any arithmetic operation, but does involve communication in a distributed memory parallel implementation. Using the restriction operator, we define the subdomain nonlinear function $F_i : R^{3n} \to R^{3n_i}$ as

$$F_i = R_i F.$$

We next define the subdomain mapping functions, which in some sense play the role of subdomain preconditioners. For any given $x \in R^{3n}$, $T_i(x) : R^{3n} \to R^{3n_i}$ is defined as the solution of the following subspace nonlinear systems,

$$F_i(x - R_i^T T_i(x)) = 0, \text{ for } = 1, ..., N. \tag{2.4}$$

Throughout this paper, we always assume that (2.4) is uniquely solvable. Using the subdomain mapping functions, we introduce a new global nonlinear function,

$$\mathcal{F}^{(1)}(x) = \sum_{i=1}^{N} R_i^T T_i(x), \tag{2.5}$$

which we refer to as the nonlinearly preconditioned $F(x)$. The one-level additive Schwarz inexact preconditioned Newton algorithm (ASPIN(1)) is defined as: Find the solution x^* of (2.3) by solving the nonlinearly preconditioned system,

$$\mathcal{F}^{(1)}(x) = 0, \tag{2.6}$$

using INB with an initial guess $x^{(0)}$. As shown in [5, 11], an approximation of the Jacobian of $\mathcal{F}^{(1)}$ takes the form

$$\mathcal{J}^{(1)}(x) \approx \sum_{i=1}^{N} J_i^{-1} J(x), \tag{2.7}$$

where J is the Jacobian of the original function $F(x)$ and $J_i = R_i J {R_i}^T$.

2.3. A parallel linear coarse component for the nonlinear preconditioner. The one-level ASPIN is robust, but not linearly scalable with respect to the number of processors. Some coarse preconditioner is required to couple the subdomain preconditioners. One such coarse preconditioner is proposed and tested in [6] and [14]. The nonlinear coarse system is obtained by the discretization of original nonlinear partial differential equations on a coarse mesh. Although, in general, solving the coarse systems is easier than the fine systems, a Newton-Krylov-Schwarz method sometimes is not good enough to converge the coarse system. Therefore, ASPIN(1) is used to solve the coarse system in [6, 14]. To evaluate the coarse function at certain point, one needs to solve a set of nonlinear systems of equations. Although the ASPIN(1) based coarse solver provides good mathematical properties, such as helping speed up the convergence of the linear iterative method, the computational cost to solve many coarse systems is usually high in practice. Numerical experiments [14] show that the ASPIN(1) based coarse solver work fine only for the moderate number of processors, for the large number of processors, a more efficient coarse solver is needed.

Here we introduce a new coarse system, which is linear, and the system is constructed by a linearization of the nonlinear coarse system mentioned above, using a Taylor approximation. The coarse function evaluation only requires the solution of a linear system, and hence the computational cost is reduced considerably. More precisely, we assume there exists a finite element mesh $\mathcal{T}^H$ covering the domain Ω. The two meshes $\mathcal{T}^H$ and $\mathcal{T}^h$ do not have to be nested. For the purpose of parallel computing, the coarse mesh is partitioned into non-overlapping subdomains $\{\Omega_i^H\}$ and overlapping subdomains $\{\Omega_i^{H,\delta}\}$. The corresponding sets of mesh points are denoted by $\{\mathcal{T}_i^H\}$, and $\{\mathcal{T}_i^{H,\delta}\}$. For the simplicity of our software implementation, we assume a non-overlapping partition to be *nested.* In other words, we must have

$$\Omega_i^h = \Omega_i^H$$

for $i = 1, \ldots, N$, even though the corresponding sets of mesh points do not have to be nested; i.e.,

$$\mathcal{T}_i^h \neq \mathcal{T}_i^H.$$

This also means that the same number of processors is used for both the fine and coarse mesh problems. If the overlap is taken into account, in general,

$$\Omega_i^{h,\delta} \neq \Omega_i^{H,\delta}, \quad \text{and} \quad \mathcal{T}_i^{h,\delta} \neq \mathcal{T}_i^{H,\delta}.$$

As in the fine mesh case, we can also define the restriction and extension operators R_i^c and $(R_i^c)^T$ for each coarse subdomain. On the coarse mesh $\mathcal{T}^H$, we can define finite element subspaces similar to the ones defined on the fine meshes, and discretize the original Navier-Stokes equations to obtain a nonlinear system of equations,

$$F^c(x_c^*) = 0 \tag{2.8}$$

We assume that the coarse solution x_c^* of (2.8) is determined through a pre-processing step. We denote J^c as the Jacobian matrix of the coarse mesh function F^c. Similar to the fine mesh, on the coarse subdomains, we obtain the coarse Jacobian submatrices

$$J_i^c = (R_i^c) J^c (R_i^c)^T, i = 1, \ldots, N.$$

We next define the coarse-to-fine and fine-to-coarse grid transfer operators. Let $\{\phi_j^H(x), j = 1, \ldots, m\}$ be the finite element basis functions on the coarse grid, where m is the total number of coarse grid points in $\mathcal{T}^H$. We define an $3n \times 3m$ matrix I_H^h, the coarse-to-fine extension matrix, as

$$I_H^h = \left[E_1\, E_2 \cdots E_n\right]^T,$$

where the block matrix E_i of size $3 \times 3m$ is given by

$$E_i = \begin{bmatrix} (e_H^h)_i & 0 & 0 \\ 0 & (e_H^h)_i & 0 \\ 0 & 0 & (e_H^h)_i \end{bmatrix}$$

and the row vector $(e_H^h)_i$ of length m is given by

$$(e_H^h)_i = \left[\phi_1^H(x_i), \phi_2^H(x_i), \ldots \phi_m^H(x_i)\right], \; x_i \in \mathcal{T}^h$$

for $i = 1, \ldots, n$. A global coarse-to-fine extension operator I_H^h can be defined as the transpose of I_h^H.

To define the coarse function $T_0 : R^{3n} \to R^{3n}$, we introduce a projection $T^c : R^{3n} \to R^{3m}$ as the solution of the linearize coarse system

$$F^c(x_c^*) + J^c(x_c^*)(T^c(x) - x_c^*) = I_h^H F(x), \tag{2.9}$$

for any given $x \in R^{3n}$. Note that the left hand side of (2.9) is a first order Taylor approximation of $F^c(x)$ at the exact coarse mesh solution, x_c^*.

Since $F^c(x_c^*) = 0$, we rewrite (2.9) as

$$T^c(x) = x_c^* + (J^c(x_c^*))^{-1} I_h^H F(x),$$

provided that $J^c(x_c^*)$ is nonsingular. It is easy to see that $T^c(x^*)$ can be computed without knowing the exact solution x^* of F, and $T^c(x^*) = x_c^*$. Then the coarse function can be defined as

$$T_0(x) = I_H^h(T^c(x) - T^c(x^*)) = I_H^h(J^c(x_c^*))^{-1} I_h^H F(x)$$

and its derivative is given by

$$\frac{\partial T_0(x)}{\partial x} = I_H^h(J^c(x_c^*))^{-1} I_h^H J(x). \tag{2.10}$$

We introduce a new nonlinear function

$$\mathcal{F}^{(2)}(x) = T_0(x) + \sum_{i=1}^{N} R_i^T T_i(x),$$

and combining (2.10) and (2.7), we obtain an approximation of Jacobian of $\mathcal{F}^{(2)}$ in the form

$$\mathcal{J}^{(2)}(x) \approx \left\{ I_H^h(J^c(x_c^*))^{-1} I_h^H + \sum_{i=1}^{N} \left[R_i^T (J_i(x))^{-1} R_i \right] \right\} J(x).$$

The two-level additive Schwarz preconditioned inexact Newton algorithm with a linear coarse solver (ASPIN(2)) is defined as: Find the solution x^* of (2.3) by solving the nonlinearly preconditioned system

$$\mathcal{F}^{(2)}(x) = 0, \tag{2.11}$$

using INB with an initial guess $x^{(0)}$. Details of ASPIN(2) is given below.

Let $x^{(0)}$ be an initial guess and $x^{(k)}$ the current approximate solution. Then a new approximate solution $x^{(k+1)}$ can be computed by the ASPIN(2) algorithm as follows:

Step 1: Evaluate the nonlinear residual $\mathcal{F}^{(2)}(x)$ at $x^{(k)}$ through the following steps:

1. Find $w_0^{(k)}$ by solving the linearize coarse mesh problem

   $$J^c(x_c^*) z_c = I_h^H F(x^{(k)}) \tag{2.12}$$

   using a Krylov-Schwarz method with a left preconditioner, $P^{-1} = \sum_{i=1}^{N} (R_i^c)^T (J_i^c)^{-1} R_i^c$ and the initial guess $z_c = 0$.
2. Find $w_i^{(k)} = T_i(x^{(k)})$ by solving in parallel, the local nonlinear systems

   $$G_i(w) \equiv F_i(x_i^{(k)} - w) = 0 \tag{2.13}$$

   using Newton method with backtracking and the initial guess $w = 0$.

3. Form the global residual

$$\mathcal{F}^{(2)}(x^{(k)}) = I_H^h w_0^{(k)} + \sum_{i=1}^{N} R_i^T w_i^{(k)}.$$

Step 2: Check the stopping condition on $||\mathcal{F}^{(2)}(x^{(k)})||_2$. If $||\mathcal{F}^{(2)}(x^{(k)})||_2$ is small enough, stop, otherwise, continue.

Step 3: Evaluate pieces of the Jacobian matrix $\mathcal{J}^{(2)}(x)$ of the preconditioned system that are needed in order to multiply (2.14) below with a vector in the next step. This includes J_i and its sparse LU factorization.

$$\mathcal{J}^{(2)} \approx \left\{ I_H^h (J^c(x_c^*))^{-1} I_h^H + \sum_{i=1}^{N} \left[R_i^T (J_i(x^{(k)}))^{-1} R_i \right] \right\} J(x^{(k)}). \tag{2.14}$$

Step 4: Find an inexact Newton direction $s^{(k)}$ by solving the following Jacobian system approximately using a Krylov subspace method

$$\mathcal{J}^{(2)} s^{(k)} = \mathcal{F}^{(2)}(x^{(k)}) \tag{2.15}$$

in the sense that

$$||\mathcal{F}^{(2)}(x^{(k)}) - \mathcal{J}^{(2)}(x^{(k)}) s^{(k)}||_2 \le \eta_k ||\mathcal{F}^{(2)}(x^{(k)})||_2 \tag{2.16}$$

for some $\eta_k \in [0, 1)$.

Step 5: Scale the search direction $s^{(k)} \leftarrow \frac{s_{max}}{||s^{(k)}||_2} s^{(k)}$ if $||s^{(k)}||_2 \ge s_{max}$.

Step 6: Compute a new approximate solution

$$x^{(k+1)} = x^{(k)} - \lambda^{(k)} s^{(k)},$$

where $\lambda^{(k)}$ is a damping parameter determined by the standard backtracking procedure.

3. Numerical results. In this section, we consider a two-dimensional lid-driven cavity flow problem [9] as a benchmark for evaluating the performance of ASPIN. Particularly, we study the linear and nonlinear scalability of the methods. We use PETSc [1] for the parallel implementation and obtain all numerical results on a cluster of workstations. Only machine independent results are reported here in this paper. In our implementation, after ordering the mesh points, we number unknown nodal values in the order of u_1^h, u_2^h, and p^h at each mesh point. The mesh points are grouped subdomain by subdomain for the purpose of parallel processing. Regular checkerboard partitions are used for our experiments. The number of subdomains is always the same as the number of processors, *np*. At the fine mesh level, we use the standard backtracking technique [7] for both global and local nonlinear problems. The global nonlinear iteration is stopped if the condition $||\mathcal{F}^{(2)}(x^{(k)})||_2 \le 10^{-6} ||\mathcal{F}^{(2)}(x^{(0)})||_2$ is satisfied, and the local nonlinear iteration on each subdomain is stopped if the condition $||G_i(w_{i,l}^{(k)})||_2 \le 10^{-4} ||G_i(w_{i,0}^{(k)})||_2$ is satisfied. We use a restarted GMRES(200) for solving the global Jacobian systems (2.15). The global linear iteration is stopped if the relative tolerance $||\mathcal{F}^{(2)}(x^{(k)}) - \mathcal{J}^{(2)}(x^{(k)}) s^{(k)}||_2 \le 10^{-6} ||\mathcal{F}^{(2)}(x^{(k)})||_2$ is satisfied. During local nonlinear iterations, a direct sparse solver, LU decomposition, is employed for solving each local Jacobian system. At the coarse mesh level, we use a restarted GMRES(200) method with a left overlapping Schwarz preconditioner to solve the coarse systems (2.12). The stopping criterion for the coarse mesh problem is that the condition $||I_h^H F(x^{(k)}) - J^c(x_c^*) z_c||_2 \le 10^{-10} ||I_h^H F(x^{(k)})||_2$ is satisfied. We use the overlapping size, $\delta = 2$ for both the fine and coarse systems. As suggested in [11], we include the re-scaling of the search direction $s^{(k)}$ in Step 5 if $||s^{(k)}||_2 \ge s_{max}$ to enhance the robustness of ASPIN for

Table 3.1

ASPIN(2): Varying the coarse mesh size for different values of Reynolds number. Fine mesh: 128×128. *The number of processors np=16.*

coarse mesh	$Re = 10^3$	3×10^3	5×10^3	8×10^3	10^4
Number of global nonlinear iterations					
16×16	8	11	11	14	17
20×20	9	9	11	13	14
32×32	8	9	11	10	12
40×40	8	9	9	10	11
64×64	8	10	9	11	11
Average number of global linear iterations					
16×16	58	74	94	111	122
20×20	50	66	75	89	103
32×32	45	52	59	64	68
40×40	43	50	54	60	60
64×64	42	49	52	55	65

solving incompressible flows. This step also reduces the number of line search steps since the evaluation of nonlinearly preconditioned function is expensive. All numerical results reported here are based on the optimal choice of the parameter s_{max} which results in the smallest number of global nonlinear iterations.

We first study the effect of the coarse mesh size on the global nonlinear iterations and the global linear iterations of ASPIN(2) for different values of Reynolds number. In this set of numerical experiments, all results are obtained using a fixed fine mesh 128×128 on 16 processors, and the coarse mesh size is varied from 16×16 to 80×80. Table 3.1 shows that to apply two-level methods on the moderate number of processors, the coarse mesh has to be sufficiently fine, say 40×40 in this case. For this particular case, the numbers of global nonlinear iterations, as well as global linear iterations, are not very sensitive with the increase of Reynolds number.

To study the parallel scalability of ASPIN(2) with respect to the number of processors, we use a fixed fine mesh 128×128 and a coarse mesh 40×40. For comparison purposes, we also include the results obtained using ASPIN(1). Table 3.2 shows that by adding a coarse preconditioner, not only the global linear iterations is reduced significantly as we increase the number of processors from 4 to 64, but also the global nonlinear iterations is improved especially for high Reynolds number flows. Table 3.3 shows the mesh scalability of ASPIN(2). Now we increase the fine mesh size from 128×128 to 512×512 on 64 processors, and we use a fixed coarse mesh 64×64 for all test cases. We see that the coarse mesh of size 64×64 works fine only for first two small meshes(128×128 and 256×256). The global linear iterations increase quite a bit for the largest mesh(512×512). For a fully scalable algorithm, one may need to increase the coarse mesh size as well, when the fine mesh size and the number of processors increase. The optimal coarse mesh size also depends mildly on the Reynolds number.

In the next set of numerical experiments, instead of using a zero vector as the initial guess, we employ the interpolation of the coarse mesh solution, $I_H^h x_c^*$, as the initial guess. Here we refer to the two-level ASPIN with the new initial guess as ASPIN(2'). Note that no additional cost is needed for this new selection of the initial guess since we need x_c^* for the nonlinear function evaluation anyway. However, with this small modification, we see a much better overall performance of the two-level ASPIN. We rerun the same test cases presented in Table 3.3 using ASPIN(2') and the results are shown in Table 3.4. Obviously ASPIN(2') performs better than ASPIN(2). ASPIN(2') requires only about half of the global nonlinear and linear iterations than ASPIN(2).

4. Concluding remarks. We presented a new two-level ASPIN algorithm and its application to incompressible Navier-Stokes equations. The two-level nonlinear preconditioner is constructed by using a local nonlinear overlapping Schwarz domain decomposition and a global

TABLE 3.2

ASPIN(1) and ASPIN(2): Varying the number of processors. Fine mesh size: 128×128*. Coarse grid size:* 40×40.

np	$Re = 10^3$	3×10^3	5×10^3	8×10^3	10^4
ASPIN(1)					
Number of global nonlinear iterations					
$2 \times 2 = 4$	9	10	13	19	19
$4 \times 4 = 16$	9	12	12	16	18
$8 \times 8 = 64$	10	15	14	19	19
Average number of global linear iterations					
$2 \times 2 = 4$	67	69	71	73	74
$4 \times 4 = 16$	127	128	133	137	140
$8 \times 8 = 64$	395	394	400	497	655
ASPIN(2)					
Number of global nonlinear iterations					
$2 \times 2 = 4$	9	9	11	10	12
$4 \times 4 = 16$	8	9	9	10	11
$8 \times 8 = 64$	8	9	12	12	14
Average number of global linear iterations					
$2 \times 2 = 4$	33	40	40	40	46
$4 \times 4 = 16$	43	50	54	60	60
$8 \times 8 = 64$	49	62	61	78	79

TABLE 3.3

ASPIN(2): Varying the fine mesh size for different values of Reynolds number. Coarse mesh size: 64×64*. The number of processors* $np = 64$

fine mesh	$Re = 10^3$	3×10^3	5×10^3	8×10^3	10^4
Number of global nonlinear iterations					
128×128	8	9	10	11	10
256×256	8	11	12	13	14
512×512	9	11	14	15	19
Average number of global linear iterations					
128×128	50	61	70	87	94
256×256	63	82	85	93	94
512×512	90	134	171	144	111

linear coarse solver. We obtained some encouraging numerical results for moderate number of processors. We show that the new two-level ASPIN maintains the fast convergence and robustness properties of the one-level ASPIN. In addition, if the coarse mesh size is fine enough the new algorithms provide better nonlinear and linear scalability with respect to the number of processors. To show the applicability of ASPIN for larger problems, more applications with complex geometry and using larger number of processors need to be tested in the future.

REFERENCES

[1] S. BALAY, K. BUSCHELMAN, W. D. GROPP, D. KAUSHIK, M. KNEPLEY, L. C. MCINNES, B. F. SMITH, AND H. ZHANG, *Portable, Extensible, Toolkit for Scientific Computation(PETSc) home page*, http://www.mcs.anl.gov/petsc, 2003.

[2] A. N. BROOKS AND T. J. R. HUGHES, *Streamline upwind/Petrov-Galerkin formulations for convective dominated flows with particular emphasis in the incompressible Navier-Stokes equations*, Comput. Methods Appl. Mech. Engrg., 32 (1982), pp. 199-259.

[3] W. L. BRIGGS, V. E. HENSON, AND S. F. MCCORMICK, *A Multigrid Tutorial*, SIAM, Philadelphia, 2000.

[4] X.-C. CAI, W. D. GROPP, D. E. KEYES, R. G. MELVIN, AND D. P. YOUNG, *Parallel Newton-Krylov-Schwarz algorithms for the transonic full potential equation*, SIAM J. Sci. Comput., 19 (1998), pp. 246-265.

[5] X.-C. CAI AND D. E. KEYES, *Nonlinearly preconditioned inexact Newton algorithms*, SIAM J. Sci. Comput., 24 (2002), pp. 183-200.

[6] X.-C. CAI, D. E. KEYES, AND L. MARCINKOWSKI, *Nonlinear additive Schwarz preconditioners and appli-*

Table 3.4

ASPIN(2'): Varying the fine mesh size for different values of Reynolds number. Coarse mesh size: 64×64. *The number of processor* $np = 64$.

fine mesh	$Re = 10^3$	3×10^3	5×10^3	8×10^3	10^4
Number of global nonlinear iterations					
128×128	3	4	5	5	6
256×256	3	4	5	6	7
512×512	4	5	6	7	8
Average number of global linear iterations					
128×128	26	28	30	35	36
256×256	31	34	38	43	48
512×512	48	49	51	56	60

cations in computational fluid dynamics, Int. J. Numer. Meth. Fluids, 40 (2002), pp. 1463-1470.

[7] J. Dennis and R. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, SIAM, Philadelphia, 1996.

[8] L. P. Franca and S. L. Frey, *Stabilized finite element method: II. The incompressible Navier-Stokes equation*, Comput. Methods Appl. Mech. Engrg., 99 (1992), pp. 209-233.

[9] U. Ghia, K. N. Ghia, and C. T. Shin, *High-Re solution for incompressible flow using the Navier- Stokes equations and the multigrid method*, J. Comput. Phy., 48 (1982), pp. 387-411.

[10] M. D. Gunzburger, *Finite Element Methods for Viscous Incompressible Flows*, Academics Press, New York, 1989.

[11] F.-N. Hwang and X.-C. Cai, *A parallel nonlinear additive Schwarz preconditioned inexact Newton algorithm for incompressible Navier-Stokes equations*, 2003. (submitted)

[12] F.-N. Hwang and X.-C. Cai, *Improving robustness and parallel scalability of Newton method through nonlinear preconditioning*, Proceedings of the 15th International Conference on Domain Decomposition Methods, 2003. (submitted)

[13] A. Klawonn and L. F. Pavarino, *Overlapping Schwarz method for mixed linear elasticity and Stokes problems*, Comput. Methods Appl. Mech. Engrg., 165 (1998), pp. 233-245.

[14] L. Marcinkowski and X.-C. Cai, *Parallel performance of some two-level ASPIN algorithms*, Proceedings of the 15th International Conference on Domain Decomposition Methods, 2003. (to appear)

[15] J. Nocedal and S. J. Wright, *Numerical Optimization*, Springer-Verlag, New York, 1999.

[16] J. N. Reddy and D. K. Gartling, *The Finite Element Method in Heat Transfer and Fluid Dynamics*, CRC Press, Florida, 2000.

[17] Y. Saad and M.H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsysmetric linear system*, SIAM J. Sci. Stat. Comp., 7 (1986), pp. 856-869.

[18] J. N. Shadid, R. S. Tuminaro, and H. F. Walker, *An inexact Newton method for fully coupled solution of the Navier-Stokes equations with heat and mass transport*, J. Comput. Phys., 137 (1997), pp. 155-185.

[19] B. Smith, P. Bjørstad, and W. Gropp, *Domain Decompostion: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, Combridge, 1996.