PARALLEL NEWTON-KRYLOV-SCHWARZ ALGORITHMS FOR THE TRANSONIC FULL POTENTIAL EQUATION

XIAO-CHUAN CAI*, WILLIAM D. GROPP†, DAVID E. KEYES‡, ROBIN G. MELVIN§ AND DAVID P. YOUNG¶

Abstract. We study parallel two-level overlapping Schwarz algorithms for solving nonlinear finite element problems, in particular, for the full potential equation of aerodynamics discretized in two dimensions with bilinear elements. The overall algorithm, Newton-Krylov-Schwarz (NKS), employs an inexact finite-difference Newton method and a Krylov space iterative method, with a two-level overlapping Schwarz method as a preconditioner. We demonstrate that NKS, combined with a density upwinding continuation strategy for problems with weak shocks, is robust and economical for this class of mixed elliptic-hyperbolic nonlinear partial differential equations, with proper specification of several parameters. We study upwinding parameters, inner convergence tolerance, coarse grid density, subdomain overlap, and the level of fill-in in the incomplete factorization, and report their effect on numerical convergence rate, overall execution time, and parallel efficiency on a distributed-memory parallel computer.

Key words. Full potential equation, finite elements, domain decomposition, Newton methods, Krylov space methods, overlapping Schwarz preconditioner, parallel computing.

AMS(MOS) subject classifications. 65H20, 65N30, 65N55, 65Y05, 76G25, 76H05

1. Introduction. In the past few years domain decomposition methods for linear partial differential equations, including overlapping Schwarz methods [10, 13, 14, 38], have graduated from theory into practice in many applications [18, 28, 29, 35]. In this paper, we study several aspects of the parallel implementation of a Krylov-Schwarz domain decomposition algorithm for the finite element solution of the nonlinear full potential equation of aerodynamics, extending our model studies of linear convection-diffusion problems in [5] and of linear aerodynamic design optimization problems in [34]. Newton-Krylov methods [2, 3, 15, 16, 40] are potentially well suited and increasingly popular for the implicit solution of nonlinear problems whenever it is expensive to compute or store a true Jacobian. We employ a combined algorithm, called Newton-Krylov-Schwarz, and focus on the interplay of the three nested components of the algorithm, since the amount of work done in each component affects and is affected by the work done in the others.

Newton-Krylov-Schwarz is a general purpose parallel solver for nonlinear partial differential equations and has been applied to complex multicomponent systems of compressible and reacting flows in, e.g., [7, 8, 30]. This paper is concerned with the simpler scalar problem of the full potential equation, which describes inviscid, irrotational, isentropic compressible flow. Though the full potential model is highly idealized, it remains the model of choice of external aerodynamic designers to date, because codes based thereupon offer reasonable turnaround times and in many cases high accuracy compared to state-of-the-art Navier-Stokes solvers. Though derived under the condition of isentropy, the full potential model remains useful in flows

* Department of Computer Science, University of Colorado at Boulder, Boulder, CO 80309. *cai@cs.colorado.edu*. This work was supported in part by NSF grants ASC-9457534, ASC-9217394, and ECS-9527169, by NASA grant NAG5-2218, and by NASA contract NAS1-19480 while the author was in residence at the Institute for Computer Applications in Science and Engineering.

† Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439. *gropp@mcs.anl.gov*. This work was supported by the Office of Scientific Computing, U.S. Department of Energy, under Contract W-31-109-Eng-38.

‡ Department of Computer Science, Old Dominion University, Norfolk, VA 23529-0162 and ICASE, NASA Langley Research Center, Hampton, VA 23681. *keyes@icase.edu*. This work was supported in part by NSF grants ECS-8957475 and ECS-9527169, by the State of Connecticut and the United Technologies Research Center, and by NASA contract NAS1-19480 while the author was in residence at the Institute for Computer Applications in Science and Engineering.

§ The Boeing Company, Seattle, WA 98124. *rgm4152@cfdd53.cfd.ca.boeing.com*.

¶ The Boeing Company, Seattle, WA 98124. *dpy6629@cfdd51.cfd.ca.boeing.com*.

with weak shocks, with pre-shock Mach numbers of about 1.2 or less. It can also be extended by boundary layer patching to incorporate viscous effects, by a branch cut to accommodate lift, and by source terms to simulate powered engines. In engineering practice, accurately modeling such nonideal effects in complex geometries accounts for almost all of the lines of code, but the solution of the resulting discrete equations accounts for the majority of the execution time. The lower per-cell storage and computational requirements of the potential model allow the use of grids dense enough to achieve low truncation error levels for complex geometries. The full potential equation also avoids the spurious entropy generation near stagnation often associated with Euler and Navier-Stokes codes for industrial complex geometries of interest. We justify the simply coded examples in this paper by our focus on a solution algorithm that should not require any changes other than greater irregularity in its sparse data structures to be useful in more practical settings.

With Newton's method as the outer iteration, a highly nonsymmetric and/or indefinite large, sparse Jacobian equation needs to be solved at every iteration to a certain accuracy, which is often progressively tightened in response to a falling nonlinear residual norm. The most popular family of preconditioners for large sparse Jacobians on structured or unstructured grids, incomplete factorization, is difficult to parallelize efficiently flop-for-flop in its global form. In our approach, the ILU-preconditioner for the Newton correction equations is replaced by a multi-level overlapping Schwarz preconditioner. The latter is not only scalably parallelizable (up to available granularities), but also possesses an asymptotically optimal mesh- and granularity-independent convergence rate for elliptically dominated problems. Our two-level overlapping additive Schwarz algorithm uses a non-nested coarse space. Subdomain granularity, quality of subdomain solves, coarse grid density, strategy for coarse grid solution, and inner iteration termination criteria are important factors in overall performance. We report numerical experiments on an IBM SP2 with up to 32 processors.

The outline of this paper is as follows. In §2, we briefly derive the form of the full potential equation that serves as the point of departure for the numerics. The finite element discretization and the construction of an approximate Jacobian for the full potential equation are discussed in §3. §4 is devoted to the description of the basic components of the NKS algorithm. Several parallel implementation issues are explained in §5. Numerical results are summarized in §6. Finally, we offer some general remarks on the use of NKS algorithms in §7.

2. The full potential problem. For completeness, we summarize the derivation and assumptions of the full potential equation of aerodynamics. For a more thorough development, see [24].

The equation of mass conservation in a steady state fluid flow can be written in divergence form, $\nabla \cdot (\rho v) = 0$, where $v = (v_1, v_2)^T$ is the velocity and ρ is the local density, respectively. We assume that the flow is irrotational, which implies that there exists a velocity potential Φ such that $v = \nabla\Phi$. Furthermore, the relation $\frac{p}{\rho^\gamma} = const.$ holds for isentropic flow of a perfect gas. With the above assumptions, we can integrate the inviscid momentum equations and obtain Bernoulli's equation $\frac{q^2}{2} + \frac{a^2}{\gamma-1} = const.$, where $q = (v_1^2 + v_2^2)^{1/2} = ||\nabla\Phi||_2$ is the local flow speed. The sound speed a is defined by $a^2 = dp/d\rho$, where p is the local static pressure. By means of the above relations, the five unknown fields v_1, v_2, p, a, and ρ can be eliminated in favor of a single unknown function Φ, which solves the full potential equation:

$$\nabla \cdot (\rho(\Phi)\nabla\Phi) = 0. \tag{1}$$

Two forms of this equation are standard in the literature, depending upon whether the density is referenced to a uniform freestream (at ∞) or to a stagnation point condition. We derive the freestream version as follows. From Bernoulli's equation,

$$\frac{q^2}{2} + \frac{a^2}{\gamma - 1} = \frac{q_\infty^2}{2} + \frac{a_\infty^2}{\gamma - 1}, \tag{2}$$

we have that

$$\frac{a^2}{a_\infty^2} = 1 + \frac{(\gamma-1)(q_\infty^2 - q^2)}{2a_\infty^2} = 1 + \frac{\gamma-1}{2} M_\infty^2 \left(1 - \frac{q^2}{q_\infty^2}\right), \tag{3}$$

where $M = q/a$ is the Mach number, and M_∞ is the freestream Mach number. From the definition of the sound speed and the pressure-density relation, we obtain

$$a^2 = \frac{d(c\rho^\gamma)}{d\rho} = c\gamma\rho^{\gamma-1}, \tag{4}$$

or equivalently, $(\frac{a}{a_\infty})^2 = (\frac{\rho}{\rho_\infty})^{\gamma-1}$. Therefore,

$$\rho(\Phi) = \rho_\infty \left(1 + \frac{\gamma-1}{2} M_\infty^2 (1 - \frac{\|\nabla\Phi\|_2^2}{q_\infty^2})\right)^{1/(\gamma-1)}. \tag{5}$$

Observe that while the density is positive in regions of validity, (1) may be locally hyperbolic.

Equation (1) requires boundary conditions. In this paper, we consider only subsonic farfield boundaries. Since our emphasis is on the performance of the Schwarz preconditioning, we study a symmetric nonlifting case, thus avoiding consideration of the Kutta-Joukowsky boundary condition. To keep the geometry of the domain trivial, we use a classical transpiration boundary condition on a slit to represent the airfoil. Transpiration refers to a continuously parameterized injection and removal of fluid along a portion of the boundary to create a recirculation pocket with a bounding streamline attached to the domain boundary at both ends, over which the flow of interest passes inviscidly. Transpiration is implemented as an inhomogeneous Neumann condition. A theoretical discussion of the use of transpiration boundary conditions to model displaced surfaces can be found in [26]. For the farfield boundary condition, we use Dirichlet values of the potential upstream. More sophisticated farfield conditions are possible, and are required in the case of a lifting airfoil, but these conditions are sufficient for excellent agreement of our numerical results with standard nonlifting solutions.

3. Finite element approximation. Following Boeing's TRANAIR code [43], we employ a finite element formulation of the two-dimensional full potential equation using bilinear elements. The existence, uniqueness, and regularity of the solution are not central to this paper, but have been discussed in the papers [32, 36] and references therein. Related finite element approaches for this class of full potential equations can also be found in [1, 17]. A finite volume scheme was given in [31] and a mortar element-based domain decomposition scheme was recently parallelized to high efficiency in [27].

3.1. Basic finite element scheme. The finite element problem is formulated in terms of the weak form: find $\Phi \in V^h \subset H^1(\Omega)$ such that

$$a(\Phi, \Psi) = \int_\Omega \rho(\Phi)\nabla\Phi \cdot \nabla\Psi \, d\Omega \;, \forall \Psi \in V^h.$$

We use bilinear elements on a rectangular partition of Ω denoted by $\Omega_h = \{\tau_i, i = 1, \ldots, M_h\}$. Let $\{\phi_i(x,y)\}$ be the usual nodal basis functions. The numerical solution we seek has the form $\Phi(x,y) = \sum \Phi_i \phi_i(x,y)$ and satisfies the following nonlinear algebraic equations

$$\sum_{i=1}^{M_h} \rho(\Phi(x_i^c, y_i^c)) \int_{\tau_i} \nabla\Phi\nabla\Psi \, d\Omega \;=\; 0 \tag{6}$$

for all Ψ in the test function space. Here (x_i^c, y_i^c) is the center point of the rectangle τ_i. To simplify and speed up numerical integration, we introduce certain approximations when dealing with some of the nonlinear forms. The way that we treat the local nonlinear numerical integration in (6) is like that in [43]. Let us define a system of nonlinear equations

$$F(\Phi) = \begin{pmatrix} F_1(\Phi_1, \cdots, \Phi_N, \rho(\Phi_1, \cdots, \Phi_N)) \\ \vdots \\ F_N(\Phi_1, \cdots, \Phi_N, \rho(\Phi_1, \cdots, \Phi_N)) \end{pmatrix} = 0, \tag{7}$$

where

$$F_i(\Phi, \rho(\Phi)) = \sum_{\tau \in \Omega_h} \rho(\Phi(x_\tau^c, y_\tau^c)) \int_\tau \nabla\Phi \nabla\phi_i d\Omega, \tag{8}$$

and (x_τ^c, y_τ^c) is the center point of the element τ. We construct the Jacobian matrix $J = \{J_{ij}\}$ of the nonlinear system $F = 0$, approximately, as follows. For each pair of indices i, j, we define

$$\frac{\partial F_i(\Phi)}{\partial \Phi_j} = \sum_{\tau \in \Omega_h} \int_\tau \rho(\Phi)(\nabla\phi_j \cdot \nabla\phi_i) d\Omega + \sum_{\tau \in \Omega_h} \int_\tau \frac{d\rho}{ds} \frac{\partial s}{\partial \Phi_j} (\nabla\Phi \cdot \nabla\phi_i) d\Omega,$$

where $s \equiv \|\nabla\Phi\|_2^2$. To simplify the numerical integration, the exact Jacobian value above is replaced by

$$J_{ij} = \sum_{\tau \in \Omega_h} \rho(\Phi(x_\tau^c, y_\tau^c)) \int_\tau (\nabla\phi_j \cdot \nabla\phi_i) d\Omega + \sum_{\tau \in \Omega_h} \frac{d\rho}{ds} \int_\tau \frac{\partial s}{\partial \Phi_j} (\nabla\Phi \cdot \nabla\phi_i) d\Omega, \tag{9}$$

where the value of $d\rho/ds$ is calculated at the element center point. We remark here that, because the density function ρ is not a constant, the Jacobian matrix is generally nonsymmetric and possibly indefinite. The explicit construction of the Jacobian matrix is not necessary if we use an unpreconditioned Newton-Krylov method; however, to implement a Schwarz preconditioner, explicit approximation of the Jacobian matrix is needed in each subdomain.

3.2. Density upwinding schemes. For subsonic problems, the above mentioned finite element method is sufficient; however, for transonic cases upwinding has to be introduced in the density calculation in order to capture the weak shock in the solution. The proper use of an upwinding scheme is essential both to the success of the overall approach in finding the correct location and strength of the shock and to the convergence, or the fast convergence, of the inexact Newton's method.

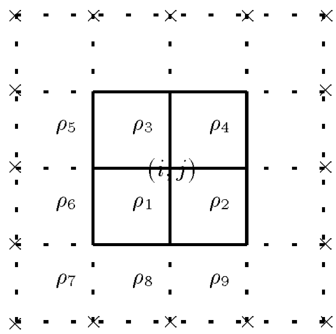

FIG. 1. *The finite element stencil.* Φ *is stored at the cell vertices and* ρ *at the cell centers.*

As mentioned earlier, density ρ is assumed to be a constant in each element, and this constant is ordinarily determined by the four values of Φ at the corners of the element, through (5). Following [25, 43], if an element is determined to be supersonic, or nearly so, its density value is replaced by $\tilde{\rho} = \rho - \mu V \cdot \nabla_- \rho$, where V is the normalized element velocity and $\nabla_- \rho$ is an upwind undivided difference. For example, with reference to Fig. 1, if $V = (V_x, V_y)$ and $V_x, V_y > 0$ in the element marked with ρ_1, then $\tilde{\rho}_1 = \rho_1 - \mu(V_x(\rho_1 -$

$\rho_6) + V_y(\rho_1 - \rho_8))$. Here μ is the element switching function, $\mu = \nu_0 \max\{0, 1 - M_c^2/M^2\}$, where M is the element Mach number, M_c is a pre-selected cutoff Mach number chosen to introduce dissipation just below Mach 1.0, and ν_0 is a constant usually set to something between 1.0 and 3.0 to increase the amount of dissipation in the supersonic elements. Parameters M_c and ν_0 may be varied to advantage between Newton steps in problems with shocks. Roughly speaking, M_c controls the spatial extent of the upwinding; as it drops below 1.0 upwinding is triggered in a greater number of subsonic (but nearly sonic) cells. M_c, ν_0, and V together control the amount of the upwinding in a triggered cell. A low M_c and a high ν_0 stabilize convergence but diffuse the shock. As iterations progress, M_c should approach 1.0 and ν_0 should be decreased to steepen up a shock whose location and strength has converged. A well resolved shock will take many Newton steps to settle on its correct location, whereas a diffuse shock centers quickly on this location. A carefully chosen sequence of M_c and ν_0 can considerably accelerate the Newton's convergence; more details on this "viscosity damping" can be found in [42]. Another way to control the convergence of Newton's method is through the use of iterated maximization of the switching function, as described below and apparently first discussed in [23].

For each element, μ as defined above is called the zeroth level switching function and is denoted more precisely as $\mu^{(0)}$. In Fig. 1, there is a nonzero $\mu_i^{(0)}$ for each element marked with ρ_i. The first level switching function for the element marked with ρ_1 is defined as $\mu_1^{(1)} = \max\{\mu_1^{(0)}, \ldots, \mu_9^{(0)}\}$, namely, $\mu_1^{(1)}$ is the maximum of the all the μ values in its immediate neighborhood. A $(k+1)$-level switching function is defined recursively as the maximum of the neighboring k-level μ values.

Results for $k = 2$ are reported in §6. A rather tight Mach cutoff is used, namely $M_c^2 = 0.95$, and we set ν_0 to 1.0.

We remark that large k results in greater discrete data dependency, or larger effective stencil size, in both the nonlinear function and the Jacobian. For example, if $k = 0$, the stencil contains at most 9 points (e.g., the nine mesh points immediately surrounding (i, j) in Fig. 1). If $k = 1$, then some of the "$\times$" points may join the stencil depending on the flow direction, and the stencil may contain as many as 16 points. The increase in the stencil bandwidth does not cause much of a problem in the nonlinear function evaluation, but would substantially increase the memory requirement of the Jacobian matrix, which is constructed and stored for the Schwarz preconditioner at the beginning of each Newton iteration. To keep the memory requirements small in practice, we do not calculate or store the matrix elements introduced by using the iterated switching function. Our numerical experiments show that this extra level of approximation of the Jacobian matrix does not, in fact, appreciably reduce its power as a preconditioner. This is analogous to the practice in [8] of using Jacobian blocks based on first-order upwinding to drive a second-order upwinded residual to zero, in an inexact Newton iteration. Though not much discussed in the theory of approximate Newton methods for systems arising from PDEs, such techniques are commonly applied in stationary iterations in steady-state aerodynamics codes. Especially in three space dimensions, using simplified upwinding in the Jacobian matrix dramatically reduces cost at a small expense in convergence rate degradation.

4. Newton-Krylov-Schwarz algorithms.

NKS is a family of general purpose algorithms for solving nonlinear boundary value problems of partial differential equations. In terms of software development, NKS has three components that can be handled independently. However, to achieve reasonable overall convergence, the three components have to be tuned simultaneously. We discuss these components in turn.

4.1. The matrix-free Newton method.

In this subsection, we briefly discuss the well known matrix-free inexact finite-difference Newton algorithm, and the Eisenstat-Walker forcing functions [16]. Starting from an initial guess Φ_0, which is sufficiently close to the solution, a solution of the nonlinear system (7) is sought by using an inexact Newton method: For some $\eta_k \in [0, 1)$ find s_k that satisfies

$$\|F(\Phi_k) + J(\Phi_k)s_k\| \leq \eta_k \tag{10}$$

and set $\Phi_{k+1} = \Phi_k + \lambda_k s_k$, where $\lambda_k \in (0,1)$ is determined by a line search procedure [12]. In practice, the method is insensitive to the details of the method used to determine λ_k. Much more important is nonlinear continuation in grid density, dissipation, and other parameters. The iteration is continued until convergence, typically defined in terms of a sufficiently small $\|F(\Phi_k)\|$. The vector s_k is obtained by approximately solving the linear Jacobian system $J(\Phi_k)s_k = -F(\Phi_k)$ with a Krylov space iterative method. The action of Jacobian J on an arbitrary Krylov vector w can be approximated by $J(\Phi_k)w \approx \frac{1}{\epsilon}(F(\Phi_k + \epsilon w) - F(\Phi_k))$. Finite-differencing with ϵ makes such matrix-free methods potentially more susceptible to finite word-length effects than ordinary Krylov methods. Left preconditioning of the Jacobian with an operator B^{-1} can be accommodated via

$$B^{-1}J(\Phi_k)w \approx \frac{1}{\epsilon}\left(B^{-1}F((\Phi_k + \epsilon w)) - \tilde{F}(\Phi_k)\right),$$

where $\tilde{F}(\Phi_k) = B^{-1}F(\Phi_k)$ is stored once, and right preconditioning via

$$J(\Phi_k)B^{-1}w \approx \frac{1}{\epsilon}\left(F((\Phi_k + \epsilon B^{-1}w)) - F(\Phi_k)\right). \tag{11}$$

Right preconditioning is preferable when the focus is on comparing different preconditioners *in vitro*, since the true linear residual norm that is measured as a by-product in Krylov method GMRES (see next subsection) and used in the termination test is independent of any right preconditioning. On the other hand, any left preconditioning changes this by-product residual norm. For this very reason, left preconditioning may be preferable when GMRES is applied *in vivo* as the solver for an inexact Newton method. When the preconditioning B^{-1} is of high quality, the left-preconditioned residual serves as an estimate of the error in the Newton update vector. This estimate can be employed in a termination condition. In this paper, one of our emphases is assessing preconditioner quality, and we report only right-preconditioned results.

The most expensive component of the algorithm is the solution of the linear system with the Jacobian at each Newton iteration. As discussed in Eisenstat and Walker [16], when Φ_k is far from the solution, the local linear model used in deriving the Newton method may disagree considerably with the nonlinear function itself, and it is unproductive to "over-solve" these linear systems. We tested several stopping conditions, including those discussed in [16], and found that the best choice for our problems, based on elapsed execution time for a fixed relative nonlinear residual norm reduction, is simply to set $\eta_k = 10^{-2}\|F(\Phi_k)\|_2$. In fact, even the looser $\eta_k = 10^{-1}\|F(\Phi_k)\|_2$ is sufficient for the first few Newton iterations, but not much time is saved by switching dynamically among these two already loose criteria, so we use the first throughout.

4.2. Krylov iterative methods. We use the GMRES method [37], to solve the linear system of algebraic equations, $Px = b$, where P is the matrix appearing in (11), and b is the negative of the nonlinear Newton residual vector in (7). The method begins with an initial approximate solution $x_0 \in R^n$ and an initial residual $r_0 = b - Px_0$. At the m^{th} iteration, a correction vector z_m is computed in the Krylov subspace $\mathcal{K}_m(r_0) = \text{span}\{r_0, Pr_0, \cdots, P^{m-1}r_0\}$ that minimizes the residual, $\min_{z\in\mathcal{K}_m(r_0)} \|b - P(x_0+z)\|_2$. The m^{th} iterate is thus $x_m = x_0 + z_m$. To fit the available memory, one is sometimes forced to use the k-step restarted GMRES method [37]. However, in this case neither an optimal convergence property nor even convergence is guaranteed. In our experiments, we do not need to solve the linear systems very accurately; i.e., $\eta = 10^{-2}$ in $\|b - Px_m\|_a \le \eta\|r_0\|_2$ is sufficient to capture an accurate solution to the nonlinear problem, in both subsonic and transonic cases. We do observe that, for certain maximum Krylov subspace dimensions (for example 30, in a problem with approximately 10^4 times as many discrete unknowns) and certain Mach numbers ($M_\infty = 0.8$), the restarted GMRES can never reduce the initial residual below 10^{-5}. In other words, there is no linear convergence. It is further noticed in such cases that the residual norm measured as a by-product in GMRES is no longer the same as, or even close to, the true residual norm except at the restarting points, where it is freshly updated.[1] A loose linear convergence tolerance avoids this problem by returning to the

[1] We believe, after Saad (personal communication), that this may be due to a lack of floating point commutativity in the product that expresses z_m in GMRES, namely $z_m = PV_my$, where V_m is a basis for

Newton method with a step that is far from exact. In the delicate balance between few nearly exact Newton steps with expensive inner linear solutions and many inexact Newton steps with bounded-cost inner linear solutions, we find the bottom line of overall execution time best served by bounding the inner linear work. This approach is also found most effective in the context of inviscid aerodynamics based on the primitive variable Euler equations in [8]. It deprives Newton's method of its asymptotic quadratic convergence, but provides steep linear convergence.

4.3. Two-level overlapping Schwarz preconditioners with non-nested coarse spaces. In this subsection, we discuss a two-level overlapping Schwarz preconditioner with inexact subdomain solvers and non-nested coarse grid. Let Ω be the domain of the full potential equation. We first partition the domain into nonoverlapping substructures Ω_i, $i = 1, \cdots, N$. To obtain an overlapping decomposition of the domain, we extend each subregion Ω_i to a larger region $\Omega_i^{'}$, i.e., $\Omega_i \subset \Omega_i^{'}$. Only simple box decomposition is considered in this paper: all the subdomains Ω_i and $\Omega_i^{'}$ are rectangular and are made up of integral numbers of fine mesh cells. For simplicity, we also assume that all substructures are of the same size. More precisely, the size of Ω_i, $i = 1, \cdots, N$, is $H_x \times H_y$ and the size of $\Omega_i^{'}$ is $H_x^{'} \times H_y^{'}$, where the $H^{'}$ are chosen so as to ensure a discrete overlap, denoted by $ovlp$, which is uniform in the number of fine grid cells all around the perimeter, i.e., $ovlp = (H_x^{'} - H_x)/2 = (H_y^{'} - H_y)/2$, for interior subdomains. For boundary subdomains, we simply cut off the part that is outside Ω. Fig. 3, which appears later with the definition of numerical boundary conditions, illustrates a decomposition with an overlap of three fine mesh cells.

On each extended subdomain $\Omega_i^{'}$, we construct a so-called subdomain preconditioner B_i, whose elements are calculated using (9). The density upwinding discussed earlier is used in the transonic cases. Homogeneous Dirichlet BCs are used on the internal subdomain boundary $\partial\Omega_i^{'} \cap \Omega$, and the appropriate external boundary condition is used on the physical boundary if present.

We next discuss the construction of the coarse grid and the coarse grid preconditioner. The coarse grid is built independently of the fine mesh. We cover Ω with another uniform rectangular mesh $\Omega_H = \{\tau_i^H, i = 1, \ldots, M_H\}$, and at each coarse node we introduce a bilinear finite element basis function $\Psi_i(x, y)$. The set of coarse nodes is not generally a subset of the fine mesh nodes. In other words, the discrete subspaces defined by the two meshes are generally non-nested [4]. Both coarse and fine grids cover the entire Ω, and they share the same boundary, which they both resolve exactly because of its prescribed simplicity. (The case of a multi-level Schwarz preconditioner for geometrically complex grids, in which only the finest level exactly resolves the boundary geometry, is considered in [11].) The coarse grid preconditioning matrix B_0 is defined by using formula (9) with respect to the basis functions $\{\Psi_i\}$. The coarse grid matrix arises from an independent discretization, not an agglomeration of the fine grid matrix. No upwinding is used on the coarse grid even in the transonic case. Empirically, the convergence may be slowed down if the density upwinding is used at the coarse grid, since a poorly located shock may be "resolved" and added to the fine grid solution. We do not fully understand the reason for this slowdown, and believe we are not alone in regarding the choice of a coarse grid operator for mixed elliptic-hyperbolic problems as one of the most important outstanding questions in multilevel preconditioning.

The interaction of the coarse and the subdomain preconditioners is through the interpolation and restriction operations. We define the coarse-to-fine interpolation matrix, I_H^h, as follows. Let $I_H^h = \{l_{ij}\}$ be an $M_h \times M_H$ matrix, and $l_{i,j} = \Psi_j(x_i)$, where x_i is ith fine mesh node. The fine-to-coarse restriction matrix is defined as $(I_H^h)^T$, the transpose of I_H^h. The additive Schwarz preconditioner can be written as

$$B^{-1} = I_H^h B_0^{-1} (I_H^h)^T + I_1 B_1^{-1} (I_1)^T + \cdots + I_N B_N^{-1} (I_N)^T. \quad (12)$$

Let $n_i^{'}$ be the total number of nodes in $\Omega_i^{'}$, then I_i is an $M_h \times n_i^{'}$ extension matrix that extends each vector

$\mathcal{K}_m$ and y is a coefficient vector of dimension m that satisfies a related least squares problem (see [37]). The effect seems related to drastic variations in the magnitude of successive elements of y.

defined on Ω_i' to a vector defined on the entire fine mesh by padding an $n_i' \times n_i'$ identity matrix with zero rows.

Various inexact additive Schwarz preconditioners can be constructed by replacing the matrices $B_i,\ i > 0$, in (12) with convenient and inexpensive to compute matrices, such as those obtained by using local incomplete factorizations. The coarse grid operator B_0^{-1} is always applied exactly. Some detailed comparisons of (12) with global ILU preconditioners on rather general scalar problems can be found in [5]. Experience with transonic potential problems in the Boeing TRANAIR code can be found in [41].

5. Parallel implementation issues. We implemented the family of NKS algorithms on the IBM SP2. The top-level message-passing calls are implemented through the Chameleon Package of Gropp and Smith [20], which uses the IBM MPL library.

The code is written in a hostless manner. Each processor is assigned one subdomain, and the information pertaining to the interior of the subdomain is uniquely owned by that processor and is not available to any other processors except by message passing. Following the parallel complexity study in [19], the low-storage coarse mesh information is duplicated in each of the processors. On each processor, we store the subvectors and subblocks of the Jacobian matrix associated with an extended subdomain. For the coarse-grid preconditioner, the right-hand vector is built by a parallel fine-to-coarse restriction operation. Once the right-hand vector is obtained, the coarse linear system is solved simultaneously on all of the processors. The solution is then added to the local subdomain solutions by using a parallel coarse-to-fine interpolation operation. In all the experiments that we have done, the size of the coarse linear system is so small that the CPU time spent on it is negligible.

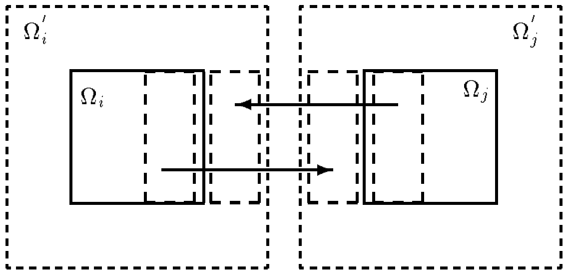

FIG. 2. *Illustration of two-way buffer copies required at each nearest-neighbor boundary. For each action of the Schwarz preconditioner on a vector the data needed in the extended regions are copied from the interior of neighboring subdomains. The amount of data moved for each processor is proportional to the area of overlap.*

The multiplication of a vector with the Schwarz preconditioner is the most expensive operation in terms of memory consumption and execution time. At the beginning of each nonlinear iteration, the Φ-dependent local and coarse grid preconditioning matrices are computed explicitly, and stored in Compressed Sparse Row (CSR) format. According to the desired type of local solver (see below), the matrices are factored, and the upper and lower triangular parts stored. The matrices for the interpolation and restriction between the coarse and fine meshes are independent of Φ, and are calculated in a preprocessing step. After the solution of each subproblem is obtained, those portions that lie within the overlapping regions (bounded by the dashed boxes in Fig. 2) are sent to neighboring subdomains to complete the summation defined in (12). The length of the message is proportional to the area of overlap.

6. Numerical results. In this section, we report some numerical results obtained on the IBM SP2 with up to 32 processors for both subsonic and transonic flows. The SP2 offers subsets of dedicated nodes through a batch scheduler. Other jobs on different dedicated subsets share the communication network, but processor allocation tends to concentrate intercommunicating processors onto independent subnetworks. We report three performance metrics for each run: (1) the total number of Newton iterations, (2) the total number of GMRES iterations, and (3) the total execution time (including the pre-processing step such as

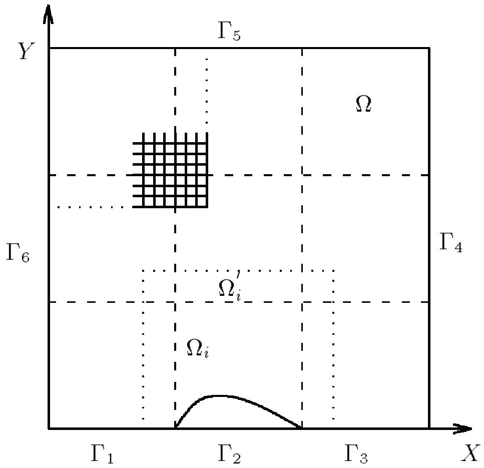

FIG. 3. *Domain Ω with an exaggerated NACA 0012 curve at the bottom. The dashed lines indicate the partition of the domain into nonoverlapping substructures, and the dotted lines indicate the overlapping subdomains. The incomplete fine mesh of solid lines illustrates an overlap of 3 subintervals. Γ_6 is the inflow, Γ_5 the freestream, and Γ_4 the outflow boundary.*

the decomposition of the mesh, the calculation of message lengths and the allocation of sparse matrices, all communication and synchronization overhead, etc.), which is an average over all processors. In [6], without length restrictions, we also report (4) the megaflop rate, which is a sum of the rates on each processor, and (5) the total communication time, which is an average over all processors (isolated out of (3)). Metrics (1) and (2) are of interest in understanding convergence rates, while (3), (4) and (5) are useful in assessing bottom-line performance and modeling scalability. The computer code was first developed on a network of workstations, and then moved to the IBM SP2, changing only a UNIX makefile.

6.1. Test problem and parameter selection. Ω is a unit-aspect ratio square partitioned into a uniform rectangular meshes up to 512×512 in size. Let q_∞, the farfield flow speed, be normalized to 1 and let $\Phi_\infty(x,y) = \int_0^x q_\infty dx$. We assume the following boundary conditions, with reference to Fig. 3:

- On the farfield boundaries Γ_4, Γ_5 and Γ_6, we assume $\Phi = \Phi_\infty$.
- On Γ_2, $\frac{\partial\Phi}{\partial y} = -\nabla\Phi_\infty \cdot (n_x, n_y)$, where $n = (n_x, n_y)$ is the unit outward normal, and where $y = f(x)$ describes the shape of the airfoil for $x \in \Gamma_2$. Once the function $f(x)$ is given, this condition becomes $\frac{\partial\Phi}{\partial y} = -q_\infty f'(x)$.
- On Γ_1 and Γ_3, we impose for symmetry the no penetration condition $\frac{\partial\Phi}{\partial n} = \frac{\partial\Phi}{\partial y} = 0$.

The functional form used for the NACA0012 geometry [39] is

$$f(x) = 0.17814(\sqrt{x} - x) + 0.10128(x(1-x)) - 0.10968x^2(1-x) + 0.06090x^3(1-x),$$

for $x \in (0,1)$. This unit interval is scaled to (1/3, 2/3) in the overall domain. The blunt leading edge of the airfoil poses a technical problem for the transpiration boundary condition, since $f'(x)$ is undefined there, so we slightly modify the function $f(x)$. The curve in the interval $[0, 0.047059]$ is replaced by a parabola with a matching function value at $x = 0$, and matching function and first derivative values at $x = 0.047059$.

A number of parameters need to be specified in the NKS algorithms. The selection of some parameters, such as the number of subdomains, is related to the granularity of the architecture, not to the equation, itself. Altogether, we have

- Switching-function parameters, in the transonic case (§3.2). The level of maximization of the switching function is set to 2, ν_0 is 1.0, and the cutoff Mach value is $M_c^2 = 0.95$.

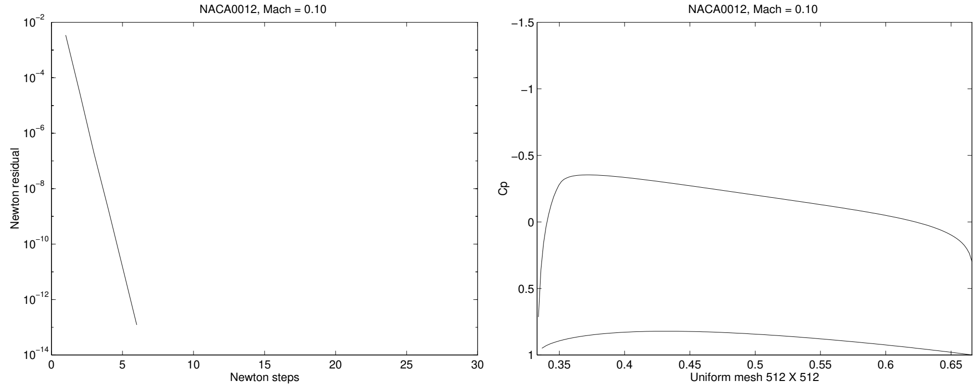

FIG. 4. *For $M_\infty = 0.1$, the left figure shows the history of the Newton residual, and the right shows the nondimensionalized coefficient of pressure curve (C_p, negative for lift) at convergence, superposed over the airfoil shape in the middle third of the domain along the symmetry boundary.*

- Finite differencing parameter, ϵ (§4.1). We find that for the nondimensional scalar full potential equation, the numerics are not very sensitive to ϵ. We simply set it to 10^{-8}, near the square root of the machine epsilon.
- Newton convergence parameters (§4.1). The initial guess is a simple interpolation of the farfield boundary condition. Nonlinear convergence is declared following a 10^{-10} relative reduction of the initial residual. The step size reduction ratio in the line search is 0.5 and the termination tolerance is 10^{-4}.
- Krylov convergence parameters (§4.2). The convergence tolerance for the linear iterative solver at each Newton iteration, η_k, is $10^{-2}\|F(\Phi_k)\|_2$. We restart GMRES at every 30th iteration.
- Number of subdomains, ns (§4.3). Since only the additive version of Schwarz is under consideration, we always set that the number of subdomains is the same as the number of processors, np, which varies from 8 (the minimum required to store the problem) to 32 (the maximum available within power-of-two configurations). (In a multiplicative algorithm [38], we would set n to np times the number of colors.)
- Overlapping size, $ovlp$ (§4.3). In fact, there are two overlapping sizes, in x and y directions. In this paper, we assume the same number of fine mesh cells, $ovlp = 1, \ldots, 5$, are extended in both directions.
- Coarse grid size (§4.3). This varies from no coarse grid (0×0) to a coarse grid with a modest number of points in each subdomain (10×11). (The coarse grid cells are square, but asymmetry in the employment of Neumann boundary conditions in the x and y directions makes the total number of gridpoints off by one.)
- Level of fill, k, in ILU (§4.3). According to our past experience with multilevel preconditioning [5] and similar experience on a industrial-grade transonic potential code [34], relatively modest fill-in is optimal for small subdomains. Intuitively, little is lost relative to the coupling already sacrificed at subdomain boundaries. However, as the local memory keeps increasing on powerful modern parallel computers, such as the IBM SP2, the size of the subdomain problems can be quite large. For large subdomain problems, low level of fill-in is no longer as effective. k varies from 0 to 5 in our experiments, then jumps discontinuously to the full band in the case of exact subdomain solves.

6.2. Observations — subsonic case. Our first test case corresponds to a subsonic problem with $M_\infty = 0.1$. The linear systems that arise fall within the elliptic theory for Schwarz [38]. It takes 6 Newton iterations to reduce the initial nonlinear residual by a factor of 10^{-10}. Because of the Krylov dimension cut-

TABLE 1

Varying the coarse grid size. *Fine mesh* 512×512*,* $M_\infty = 0.1$ *and* 0.8*. Exact LU for all subproblems,* $ovlp = 3h$*. "GMRES" is the total number of GMRES iterations incurred in all of the Newton iterations. An asterisked GMRES count in the transonic section indicates that one extra Newton step was required. "EXEC" is the execution time per processor in seconds for the entire calculation.*

np	Coarse Grid	0×0	2×3	4×5	6×7	8×9	10×11
		$M_\infty = 0.1$ (6 Newton iterations)					
	GMRES	144	81	59	53	50	51
8	EXEC	136.79	125.30	104.18	97.28	$\boxed{94.18}$	94.63
	GMRES	167	92	66	54	54	54
16	EXEC	72.18	50.34	42.10	$\boxed{39.18}$	40.95	42.32
	GMRES	227	105	72	64	53	57
32	EXEC	47.94	29.55	24.32	24.46	$\boxed{23.43}$	26.59
		$M_\infty = 0.8$ (19 Newton iterations *)					
	GMRES	814	435	359	350*	307	311
8	EXEC	757.45	548.52	490.64	498.08	$\boxed{458.75}$	466.20
	GMRES	849	483	398*	349*	320	330
16	EXEC	432.78	327.77	306.25	291.14	$\boxed{277.06}$	289.89
	GMRES	1142	636	454	387	391*	385*
32	EXEC	333.79	244.36	207.54	$\boxed{200.21}$	215.46	227.84

off, the convergence is linear; see the left panel in Fig. 4. The top portion of Table 1 shows the convergence performance for a fixed-size problem of 512×512 uniform cells with an increasing number of subdomains (not always of the same aspect ratio): $8 (= 2 \times 4)$, $16 (= 4 \times 4)$ and $32 (= 4 \times 8)$. The overlap size is fixed at $3h$. The boxed entries represent the best execution times. The density of the unnested uniform coarse grid varies from 0×0 to 10×11. Key observations from this example are as follows: (1) Even a modest coarse grid makes a significant improvement in an additive Schwarz preconditioner, especially when the number of subdomains is large. As much as 40% of the execution time can be saved when adding a 2×3 coarse grid to a no coarse grid preconditioner, for the 32-subdomain case. (2) A law of diminishing returns sets in at roughly one point per subdomain. (3) When using 8 processors, the total communication time is always less than 5% of the total computational time, however, it becomes as much as 26% when using 32 processors. (This includes synchronization delays as well as the time actually delivering the message packets from application process to application process.) Table 2 shows the effects of the overlap size. For simplicity, we fix the coarse grid to 6×7 for all test cases. The overlap size is given here in absolute terms, i.e., the distance between the boundary of the unextended subdomain and the extended subdomain, not relative to the diameter of the subdomain. All the subproblems are solved with the exact Gaussian elimination in sparse format. Since the fine mesh size is fixed, when using small number of processors, such as $np = 8$, the single processor memory requirement is substantial. In this case, increase the overlap size can indeed reduce the total number of GMRES iterations, but the reduction of the total execution time is rather limited. (The third column of Table 2 and the fourth column of Table 1 are the same method; the slight execution time differences give an idea of their reproducibility.)

In Table 3, we vary the level of fill, in ILU(k), with which the subproblems are solved. The overlap size is $3h$, and the coarse grid is 7×8. The conclusion from the tests shown is that the larger the k, the faster the method becomes; see the boxed numbers in Table 3. When using a small number of processors, like 8, the best execution time is obtained with ILU(5); compare the upper portions of Tables 1, 2 and 3. However, if the processor number is large, the optimal result can only be obtained by considering several parameters: $ovlp$, k, the coarse mesh size, and perhaps others. We have not simultaneously varied all relevant parameters

TABLE 2

Varying the overlapping size *ovlp. Fine mesh* 512×512, $M_\infty = 0.1$ *and* 0.8. *Exact LU for all subproblems. Coarse grid size* 6×7. *"GMRES" is the total number of GMRES iterations incurred in all of the Newton iterations. An asterisked GMRES count in the transonic section indicates that one extra Newton step was required. "EXEC" is the execution time per processor in seconds for the entire calculation.*

np		$ovlp = 1h$	$ovlp = 2h$	$ovlp = 3h$	$ovlp = 4h$	$ovlp = 5h$
		$M_\infty = 0.1$ (6 Newton iterations)				
8	GMRES	77	60	53	50	50
	EXEC	101.12	99.90	$\boxed{97.31}$	98.31	100.50
16	GMRES	82	61	54	49	49
	EXEC	51.77	45.46	$\boxed{44.05}$	44.42	45.21
32	GMRES	93	72	64	57	52
	EXEC	30.35	28.28	26.75	26.31	$\boxed{26.19}$
8	GMRES	435	365	350*	319*	315*
	EXEC	527.50	$\boxed{489.65}$	498.78	491.47	492.52
16	GMRES	462	377	349*	324*	303*
	EXEC	315.39	294.34	292.02	280.61	$\boxed{278.21}$
32	GMRES	627	445	387	354	344
	EXEC	251.96	207.12	199.90	$\boxed{188.99}$	189.44

to get the best results, but have presented controlled slices through parameter space for insight.

Load balancing should not be a significant issue in the dedicated-processor subsonic case. All processors have nearly the same computational load, except those which have to handle the Neumann boundary conditions. This is no longer true for the transonic calculation, when a shock resides in some of the subdomains. See §6.4.

6.3. Observations — transonic case. The first and most important observation is that without a proper upwinding discretization, all three components of NKS can fail.

Fig. 6 shows the convergence history in terms of the C_p curves. We note that it takes only 4 to 5 iterations for the Newton's method to establish the neighborhood of the shock, but another 15 or so iterations to move it to the exact location. Mach contours at the final solution are given in Fig. 6. While the shock is setting up, the linear convergence of Newton's method is interrupted; see the left panel of Fig. 5.

The results for coarse grids of varying size are summarized at the bottom part of Table 1. The columns

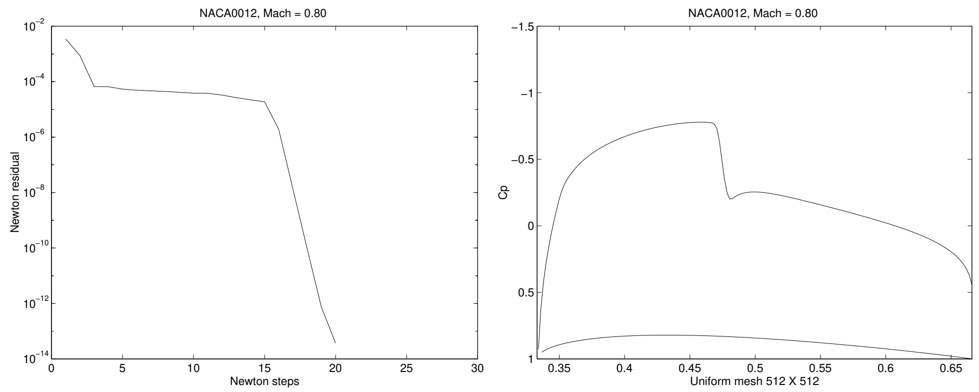

FIG. 5. *For* $M_\infty = 0.8$, *the left figure shows the history of the Newton residual, and the right shows the (upper surface)* C_p *curve at convergence.*

TABLE 3

Varying the level of ILU(k) fill-in. *Fine mesh* 512×512*,* $M_\infty = 0.1$ *and* 0.8*. Coarse grid is* 7×8*. ovlp* $= 3h$*. "GMRES" is the total number of GMRES iterations incurred in all of the Newton iterations. An asterisked GMRES count in the transonic section indicates that one extra Newton step was required and a daggered GMRES count indicates one less. "EXEC" is the execution time per processor in seconds for the entire calculation.*

np	ILU(k)	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
		$M_\infty = 0.1$ (6 Newton iterations)					
8	GMRES	307	175	127	98	84	75
	EXEC	171.46	110.07	88.26	76.11	74.28	$\boxed{71.74}$
16	GMRES	299	178	129	101	87	78
	EXEC	97.82	64.14	51.19	43.34	40.81	$\boxed{40.07}$
32	GMRES	298	179	130	104	90	82
	EXEC	64.34	41.92	32.73	28.18	26.34	$\boxed{25.57}$
		$M_\infty = 0.8$ (19 Newton iterations *†)					
8	GMRES	1638	911*	622†	547	462	424
	EXEC	863.15	523.33	382.27	364.11	335.47	$\boxed{330.76}$
16	GMRES	1693	924*	707*	531	460	423
	EXEC	526.40	316.98	255.54	209.27	195.96	$\boxed{192.16}$
32	GMRES	1746	961*	799*	596	557*	501*
	EXEC	361.81	217.29	187.37	150.59	150.29	$\boxed{145.12}$

marked 0×0 and 2×3 reveal an interesting result for a mixed elliptic-hyperbolic problem. The inclusion of a small coarse grid can reduce the total number of the linear iterations, as well as the total execution time, by a factor of 30%. An optimally chosen coarse grid size can lead to a greater savings. In Fig. 7, we overlay the convergence histories of all the linear solutions in a complete nonlinear calculation. The history in the left panel is without a coarse grid, and that in the right with a 7×8 coarse grid.

The number of linear iterations and the total execution time can be reduced even further if a proper overlap size, which is not usually very small, is used; see Table 2.

The best result, in terms of the total execution time, among all the transonic test calculations is obtained using ILU(5) as the subproblem solver; see Table 3. It takes less than $2\frac{1}{2}$ minutes on the 32-processor IBM SP2 to set up and solve the nonlinear system with more than a quarter of a million unknowns.

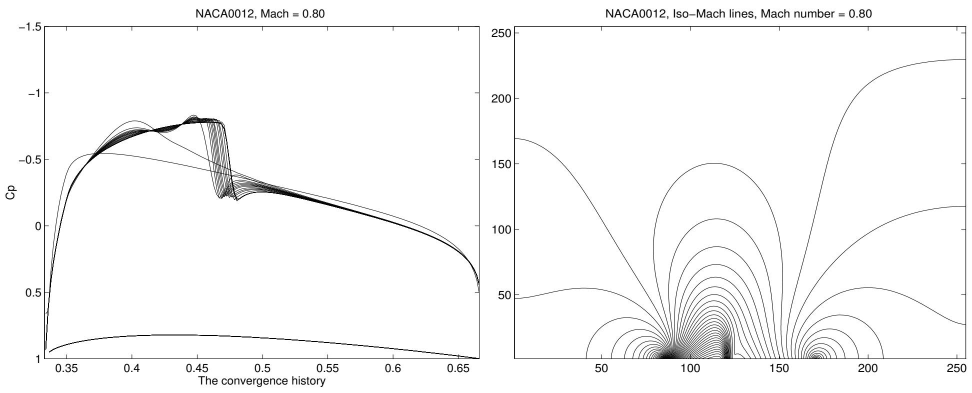

FIG. 6. *The left figure shows the convergence history of the* C_p *curves at* $M_\infty = 0.8$*. The right figure shows the Mach contours of the final solution at* $M_\infty = 0.8$*.*

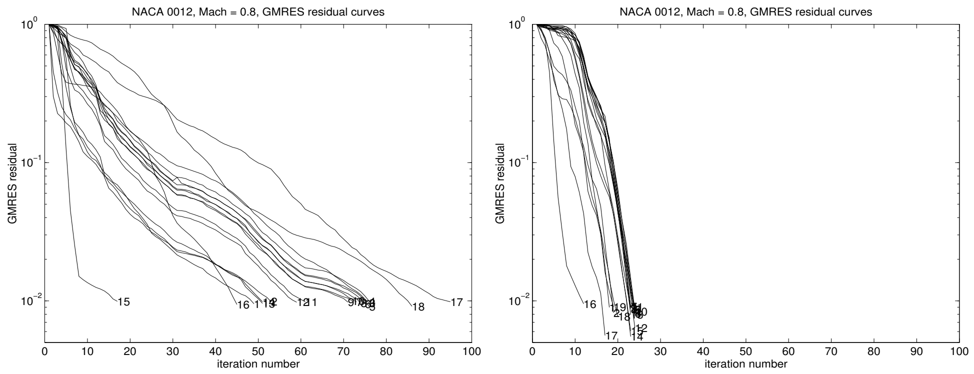

FIG. 7. *The GMRES convergence history for the entire nonlinear iteration. The left figure does not have a coarse space in the Schwarz preconditioner, the right figure contains a* 7×8 *coarse space. The number at the tail of each curve corresponds to the Newton iteration number.*

TABLE 4

Parallel efficiency. *Fine mesh* 512×512, $M_\infty = 0.1$ *and* 0.8. *Coarse grid is* 7×8. $ovlp = 3h$. *"GMRES" and "EXEC" are the total number of GMRES iterations and seconds of execution time per 6 Newton steps in the upper half of the table, and per 19 Newton steps in the lower half.*

np	GMRES	EXEC	η_{numer}	η_{impl}	η
		$M_\infty = 0.1$			
8	75	71.1	–	–	–
16	78	40.1	0.961	0.931	0.895
32	82	25.6	0.915	0.766	0.701
		$M_\infty = 0.8$			
8	424	330.1	–	–	–
16	423	192.2	1.002	0.860	0.861
32	475	137.9	0.891	0.674	0.601

6.4. Parallel efficiency. The parallel efficiency of the present algorithm-software-hardware system is encouraging, but it is useful to sort out in detail where efficiency is lost in going from 8 to 32 processors. We display the parallel performance in Table 4, whose first three columns are excerpted from the first and last columns of Table 3. The execution time data in the last column of Table 3 is the best, or nearly the best, for each Mach number and parallel granularity out of all of the parameter combinations considered, and is therefore the most meaningful from which to draw parallel efficiency conclusions, though more flattering conclusions could be drawn from runs that were performing more computation per node per communication exchange.

After the number of processors, we list the number of linear GMRES iterations per 6 Newton steps in the upper (subsonic) half of the table, and per 19 Newton steps in the lower (transonic) half of the table. Then we list the execution time, per 6 or 19 Newton steps, respectively, these being the typical number of Newton steps required to fully solve the nonlinear problem. Fig. 7 shows that, for the transonic case, the number of GMRES steps can vary significantly over the course of a complete set of Newton iterations, but the mean and the median are close.

Consider the following idealized model for the execution time of the fixed-size problem on p processors. Let $T(p)$ be the overall execution time, $I(p)$ the number of linear iterations, and $C(p)$ the average cost per iteration. (We note that in GMRES, the average cost per iteration is *not* independent of the number of iterations, because of the orthogonalization overhead of later Krylov vectors is greater than earlier vectors, but we assume that the dominant cost per iteration is the parallel Schwarz preconditioning.)

We define the overall parallel efficiency as $\eta(p) = \frac{T(1)}{p \cdot T(p)}$, where $T(p) = I(p) \cdot C(p)$. Since we lack results for $p = 1$ on this problem of industrial size (512×512), we replace all efficiencies by relative efficiencies with respect to the minimum configuration of $p_1 = 8$. The overall relative parallel efficiency is therefore defined as $\eta(p_1 \to p) = \frac{p_1 \cdot T(p_1)}{p \cdot T(p)}$. Note that this definition differs from the standard one in that the algorithm adapts to the number of processors employed, so that each processor processes exactly one subdomain. We believe this to be the most useful parallel efficiency measure, though it is not necessarily the most flattering at high granularity. We sort it out further as follows. The numerical efficiency, a measure of the robustness of the preconditioning with respect to increasing granularity, is defined as $\eta_{numer}(p_1 \to p) = \frac{I(p_1)}{I(p)}$. The implementation efficiency is the remaining factor, $\eta_{impl}(p_1 \to p) = \frac{p_1 \cdot C(p_1)}{p \cdot C(p)}$, so that $\eta(p_1 \to p) = \eta_{numer}(p) \times \eta_{impl}(p)$.

The numerical efficiency is nearly 90% or above for all cases — that is, the convergence rate of the preconditioned linear system hardly degrades with increasing parallel granularity. Approximately 13 GMRES steps are required for each subsonic Jacobian system and approximately 23 GMRES steps for each transonic Jacobian system. This insensitivity to granularity for a multilevel preconditioned operator is predicted by the Schwarz theory for the subsonic case, and seems to be a fortunate consequence of the relatively confined supersonic pocket of flow in the transonic case.

The implementation efficiency accounts for the most significant factor of overall efficiency decline. The difference between the subsonic and transonic implementation efficiencies at high granularity can be attributed to load imbalance, since the cells requiring upwinding are concentrated into a small number of processors. (A more sophisticated dynamic mapping algorithm could address this problem, but this is beyond present scope.) The subsonic degradation of 76% in going from 4 to 32 nodes is identified as the chief remaining loss. Redundant work and higher communication-to-computation ratio in the overlap regions, which account for a steadily increasing fraction of all points in a fixed-size problem explain the majority of this efficiency loss, which would disappear in a scaled problem with fixed-size subdomains on each processor.

TABLE 5

Sequential comparison *of the additive Schwarz preconditioner(OSM) with the global ILU(k), $k = 0, \ldots, 5$, preconditioners on a single processor Sun workstation. The fine mesh is 256×256. The specifications of OSM are: 8 subdomains, $3h$ overlap, 7×8 coarse grid, and ILU(5) as the subdomain solver. GMRES and EXEC are as in Tables 1–3. MEM is the total memory needed to store the preconditioning matrix in Megabytes.*

	OSM	ILU(0)	ILU(1)	ILU(2)	ILU(3)	ILU(4)	ILU(5)
	$M_\infty = 0.1$ (6 Newton iterations)						
GMRES	48	509	280	195	148	119	103
EXEC	$\boxed{496.29}$	2754.54	1599.84	1178.55	970.72	817.42	771.04
MEM(MB)	24.22	10.58	10.57	13.65	16.83	20.12	22.87
Mflop/s	32.12	6.68	6.48	6.19	6.04	5.85	5.40
	$M_\infty = 0.8$ (11 Newton iterations *)						
GMRES	136	1391*	666*	464	328	268	217
EXEC	$\boxed{1240.18}$	7766.42	3826.05	2848.73	2278.81	1761.89	1526.82
MEM(MB)	24.40	10.64	10.63	13.75	16.95	20.26	23.06
Mflop/s	36.57	6.53	6.48	6.20	5.63	5.99	5.79

6.5. Sequential comparison with global ILU(k) preconditioners. The results of this section establish Schwarz preconditioning as numerically attractive and reasonably parallel efficient, but it is natural to ask whether its utility is limited to distributed memory implementations of Newton-Krylov methods. To satisfy curiosity on this point, we conclude with tests of Schwarz preconditioning against the popular global ILU(k), $k = 0, \dots 5$, family of preconditioners on a non-dedicated single-processor SUN SPARCstation with 512MB of memory. The results are summarized in Table 5. Because of the overlap and the coarse solve, the Schwarz preconditioner needs more memory, even if all subdomain problems are solved inexactly with ILU(5), than the other global ILU(k) preconditioners. On the other hand, Schwarz outperforms all the global solvers in terms of total GMRES iteration count and the total execution time. Part of the reason for the fine performance of the Schwarz method is the much higher uni-processor megaflop/s rating, which is presumably related to much improved cache locality.

7. Concluding remarks. We have investigated computationally the effectiveness of Newton-Krylov-Schwarz methods applied to the full potential equation of aerodynamics in some simplified situations in two space dimensions. Best performance is obtained with modest overlap, a modest coarse grid (one or two points per processor), modest-to-generous fill in the subdomain ILU preconditioners, and uniformly loose convergence tolerances on the Krylov iterations within each Newton step. For subsonic problems, the theoretically expected performance of the method is essentially achieved. For the transonic case, the numerics are more encouraging than existing theory. Overall computation time is approximately six times greater for the transonic than for the subsonic case, with current upwinding strategies. This can be factored into a three-fold increase in the number of Newton steps in the transonic case, and a two-fold increase in the number of Krylov iterations per Newton step.

Two strategies that should be employed on more nonlinearly taxing problems that we have not considered here are mesh sequencing and pseudo-transient continuation. Their purpose is to deliver an initial iterate for the steady-state form of Newton's method employed in this paper that is already in the local domain of convergence on the finest grid. (Observe, for instance, that the number of Newton steps required on $M_\infty = 0.8$ problem the 256×256 grid in Table 5, is roughly half that of the corresponding problem on the 512×512 grid in Table 3. If the shock is correctly located on a (relatively) coarse grid, the plateau of Fig. 5 will be diminished on a finer grid that is initialized from the coarse grid solution.) Our rapid turnaround times for two-dimensional problems artificially deemphasize the importance of these strategies in large, complex nonlinear problems. In addition to globalizing the Newton convergence, continuation strategies tend to improve the linear conditioning of the intermediate problems, and are therefore potentially useful even in problems (such as ours) for which simple initial guesses on the finest grid lead to convergence.

The broadest motivation for Newton-Krylov-Schwarz methods is the need to solve large-scale problems with complex discretizations on distributed-memory systems with limited memory per node. The matrix-free aspect of Newton permits shortcuts in Jacobian formation storage while the domain decomposition aspect of Schwarz leads to load-balanced data-to-memory maps that render communication subdominant in the preconditioning. The amount of work done in the Krylov iteration can be adjusted to produce an overall method with the best balance between the nested components.

REFERENCES

[1] H. Berger, G. Warnecke, and W. Wendland, *Finite elements for transonic potential flows*, TR No. 7, Mathematisches Institut, Universität Stuttgart, 1988.

[2] P. N. Brown and Y. Saad, *Hybrid Krylov methods for nonlinear systems of equations*, SIAM J. Sci. Stat. Comput., 11 (1990), pp. 59–71.

[3] P. N. Brown and Y. Saad, *Convergence theory of nonlinear Newton-Krylov algorithms*, SIAM J. Optimization, 4 (1994), pp. 297–330.

[4] X.-C. Cai, *The use of pointwise interpolation in domain decomposition methods with non-nested meshes*, SIAM J. Sci. Comput., 16 (1995), pp. 250–256.

[5] X.-C. Cai, W. D. Gropp, and D. E. Keyes, *A comparison of some domain decomposition and ILU preconditioned iterative methods for nonsymmetric elliptic problems*, Numer. Lin. Alg. Applics, 1 (1994), pp. 477–504.

[6] 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*, ICASE TR 96-39, Inst. for Comput. Applics. in Sci. and Eng., NASA Langley (1996)

[7] X.-C. Cai, W. D. Gropp, D. E. Keyes, and M. D. Tidriri, *Newton-Krylov-Schwarz methods in CFD*, in *Proceedings of the Intl. Workshop on Numerical Methods for the Navier-Stokes Equations*, F. Hebeker and R. Rannacher, eds., Notes on Numerical Fluid Mechanics, Vieweg Verlag, Braunschweig (1994), pp. 17–30.

[8] X.-C. Cai, D. E. Keyes, and V. Venkatakrishnan, *Newton-Krylov-Schwarz: An implicit solver for CFD*, in "Proc. of the Eighth Intl. Conference on Domain Decomposition Methods in Science and Engineering" (R. Glowinski et al., eds.), Wiley, New York, 1996 (to appear).

[9] X.-C. Cai and O. Widlund, *Domain decomposition algorithms for indefinite elliptic problems*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 243–258.

[10] T. F. Chan and T. Mathew, *Domain decomposition algorithms*, Acta Numerica (1994) pp. 61-143.

[11] T. F. Chan and B. F. Smith, *Domain decomposition and multigrid algorithms for elliptic problems on unstructured meshes*, in *Proc. of the Seventh Intl. Symp. on Domain Decomposition Methods in Science and Engineering* (D. Keyes and J. Xu, eds.), AMS, Providence, 1995, pp. 175–189.

[12] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall, NJ, 1983.

[13] M. Dryja, B. F. Smith, and O. B. Widlund, *Schwarz analysis of iterative substructuring algorithms for problems in three dimensions*, SIAM J. Numer. Anal., 31 (1994), pp. 1662–1694.

[14] M. Dryja and O. B. Widlund, *Towards a unified theory of domain decomposition algorithms for elliptic problems*, in Third Intl. Symp. on Domain Decomposition Methods for Partial Differential Equations, held in Houston, Texas, March 20-22, 1989, T. Chan, R. Glowinski, J. Périaux, and O. Widlund, eds., SIAM, Philadelphia, PA, 1990, pp. 3–21.

[15] S. C. Eisenstat and H. F. Walker, *Globally convergent inexact Newton methods*, SIAM J. Optimization, 4 (1994), pp. 393–422.

[16] S. C. Eisenstat and H. F. Walker, *Choosing the forcing terms in an inexact Newton method*, SIAM J. Sci. Comput., 17 (1996), pp. 16-32.

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

[18] R. Glowinski et al., eds., *Proc. of the Eighth Intl. Symp. on Domain Decomposition Methods in Science and Engineering*, Wiley, New York, 1996 (to appear).

[19] W. D. Gropp and D. E. Keyes, *Domain decomposition on parallel computers*, Impact of Comp. in Sci. and Eng., 1 (1989), pp. 421–439.

[20] W. D. Gropp and B. F. Smith, *Users Manual for the Chameleon Parallel Programming Tools*, ANL-93/23, Argonne National Laboratory, 1993.

[21] ——, *Simplified Linear Equation Solvers Manual*, ANL-93/8, Argonne National Laboratory, 1993.

[22] ——, *Users Manual for KSP: Data-Structure-Neutral Codes Implementing Krylov Space Methods*, ANL-93/30, Argonne National Laboratory, 1993.

[23] W. G. Habashi and M. M. Hafez, *Finite element solutions of transonic flow problems*, AIAA J., 20 (1982), pp. 1368–1376.

[24] C. Hirsch, *Numerical Computation of Internal and External Flows*, 2 vols., Wiley, New York, 1990.

[25] T. L. Holst and W. F. Ballhaus, *Fast, conservative schemes for the full potential equation applied to transonic flows*, AIAA J., 17 (1979), pp. 145–152.

[26] W. P. Huffman, R. G. Melvin, D. P. Young, F. T. Johnson, J. E. Bussoletti, M. B. Bieterman, and C. L. Hilmes, *Practical design and optimization in computational fluid dynamics*, AIAA Paper 93-3111, July 1993.

[27] Y. Iliash, Y. Kuznetsov, and Y. Vassilevski, *Efficient Parallel Solution of Potential Flow Problems on Nonmatching Grids*, Proc. of ECCOMAS '96, Wiley, New York, 1996 (to appear).

[28] D. E. Keyes and J. Xu, eds., *Proc. of the Seventh Intl. Symp. on Domain Decomposition Methods in Science and Engineering*, AMS, Providence, 1995.

[29] D. E. Keyes, Y. Saad, and D. G. Truhlar, eds., *Domain-based Parallel and Problem Decomposition Methods in Science and Engineering*, SIAM, Philadelphia, 1995.

[30] D. A. Knoll, P. R. McHugh, and D. E. Keyes, *Newton-Krylov methods for low Mach number compressible combustion*, in "Proc. of the 12th AIAA Computational Fluid Dynamics Conference" (San Diego, June 1995), AIAA Paper 95-1672 and AIAA J., 34 (1996), pp. 961–967.

[31] C. Liu and S. F. McCormick, *Multigrid, elliptic grid generation and the fast adaptive composite grid method for solving transonic potential flow equations*, in Multigrid Methods: Theory, Applications, and Supercomputing, S. F. McCormick, ed., Lecture Notes in Pure and Appl. Math., 110, Marcel Dekker, New York, 1988, pp. 365-387.

[32] J. Mandel and J. Nečas, *Convergence of finite elements for transonic potential flows*, SIAM J. Numer. Anal., 24 (1987), pp. 985–997.

[33] J. A. Meijerink and H. A. Van der Vorst, *Guidelines for the usage of incomplete decompositions in the solving sets of linear equations as they occur in practical problems*, J. Comput. Phys., 44 (1981) pp. 134–155.

[34] R. G. Melvin, D. P. Young, D. E. Keyes, C. C. Ashcraft, M. B. Bieterman, C. L. Hilmes, W. P. Huffman, and F. T. Johnson, *A two-level iterative method applied to aerodynamic sensitivity calculations*, BCSTECH-94-047, Boeing Computer Services, December 1994.

[35] A. Quarteroni et al., eds., *Proc. of the Sixth Intl. Symp. on Domain Decomposition Methods in Science and Engineering*, AMS, Providence, 1994.

[36] R. Rannacher, *On the convergence of the Newton-Raphson method for strongly nonlinear elliptic problems*, in Nonlinear Computational Mechanics, P. Wriggers and W. Wagner, eds., Springer-Verlag, 1991.

[37] Y. Saad and M. H. Schultz, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Stat. Comp., 7 (1986), pp. 865–869.

[38] B. F. Smith, P. E. Bjørstad, and W. D. Gropp, *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, 1996.

[39] S. Ta'asan, G. Kuruvila, and M. D. Salas, *Aerodynamic Design and Optimization in One Shot*, AIAA Paper 92-0025.

[40] L. B. Wigton, N. J. Yu, and D. P. Young, *GMRES acceleration of computational fluid dynamics codes*, AIAA Paper 85-1494.

[41] D. P. Young, C. C. Ashcraft, R. G. Melvin, M. B. Bieterman, W. P. Huffman, T. F. Johnson, C. L. Hilmes, and J. E. Bussoletti, *Ordering and incomplete factorization issues for matrices arising from the TRANAIR CFD code*, BCSTECH-93-025, Boeing Computer Services, 1993.

[42] D. P. Young, R. G. Melvin, M. B. Bieterman, F. T. Johnson, and S. S. Samant, *Global convergence of inexact Newton methods for transonic flow*, Int. J. Numer. Meths. Fluids, 11 (1990), pp. 1075–1095.

[43] D. P. Young, R. G. Melvin, M. B. Bieterman, F. T. Johnson, S. S. Samant, and J. E. Bussoletti, *A locally refined rectangular grid finite element methods: Application to computational fluid dynamics and computational physics*, J. Comput. Phys., 92 (1991), pp. 1–66.