JOURNAL OF COMPUTATIONAL PHYSICS **137,** 155–185 (1997)
ARTICLE NO. CP975798

An Inexact Newton Method for Fully Coupled Solution of the Navier–Stokes Equations with Heat and Mass Transport[1]

John N. Shadid,* Ray S. Tuminaro,* and Homer F. Walker†,[2]

**Sandia National Laboratories, MS 1111, P.O. Box 5800, Albuquerque, New Mexico 87185-1111;*
†Department of Mathematical Sciences, Worcester Polytechnic Institute,
Worcester, Massachusetts 01609-2280
E-mail: jnshadi@cs.sandia.gov; tuminaro@cs.sandia.gov; walker@wpi.edu

Received January 24, 1997

The solution of the governing steady transport equations for momentum, heat and mass transfer in flowing fluids can be very difficult. These difficulties arise from the nonlinear, coupled, nonsymmetric nature of the system of algebraic equations that results from spatial discretization of the PDEs. In this manuscript we focus on evaluating a proposed nonlinear solution method based on an inexact Newton method with backtracking. In this context we use a particular spatial discretization based on a pressure stabilized Petrov-Galerkin finite element formulation of the low Mach number Navier–Stokes equations with heat and mass transport. Our discussion considers computational efficiency, robustness and some implementation issues related to the proposed nonlinear solution scheme. Computational results are presented for several challenging CFD benchmark problems as well as two large scale 3D flow simulations. © 1997 Academic Press

Key Words: Navier–Stokes equations; inexact Newton methods; Newton iterative (truncated Newton) methods; Newton–Krylov methods; forcing terms; GMRES.

[1] This work was partially funded by the United States Department of Energy, Mathematical, Information, and Computational Sciences Division, and was carried out at Sandia National Laboratories operated for the U.S. Department of Energy under Contract DE-ALO4-94AL85000.

[2] The work of this author was supported in part through a summer visit to Sandia National Laboratories and also in part by United States Department of Energy Grant DE-FG03-94ER25221 and National Science Foundation Grant DMS-9400217, both with Utah State University.

1. INTRODUCTION

We consider the simulation of fluid flow governed by the steady transport equations for momentum, heat, and mass transfer. Discretization of these equations gives rise to a system of nonlinear algebraic equations, the numerical solution of which can be very challenging. In most nontrivial calculations, the solution process is computationally intensive and requires sophisticated algorithms to cope with high nonlinearity, strong PDE coupling, and a large degree of nonsymmetry.

Newton's method is a potentially attractive nonlinear solution method because of its ability to address fully the coupling of the variables. In addition, it enjoys rapid (typically q-quadratic) convergence[3] near a solution that is not hindered by bad scaling of the variables. However, the implementation of Newton's method in the context of interest here involves special considerations. Determining steps of Newton's method requires the solution of linear systems, and iterative linear algebra methods are typically preferred for this. Consequently, obtaining exact solutions of these systems is infeasible, and the appropriate method is an *inexact Newton method* [2]. (See Section 2 below for a precise description.)

Our primary interest is in evaluating the effectiveness of a proposed inexact Newton method on these difficult fluid flow problems. In formulating this method, we have given particular consideration to implementation issues that affect robustness and efficiency. The primary mechanism for enhancing robustness is a *backtracking* (*linesearch, damping*) technique that shortens steps as necessary to ensure adequate decrease in the residual of the nonlinear system. A feature that is critical to efficiency and which can improve robustness as well, is the use of nonlinear residual information in determining the accuracy with which the linear subproblems are solved. That is, the accuracy required in solving the linear subproblems varies as the nonlinear algorithm proceeds, and this accuracy requirement is based on how well the residual of the linear system reflects the behavior of the nonlinear residual. We demonstrate in this paper that this scheme often drastically improves computational time and in some cases can help improve robustness. In addition, we have experimented with several other optimization techniques (e.g., trust regions, steepest-descent directions) in conjunction with the inexact Newton scheme. While these algorithmic enhancements may offer advantages over the basic inexact Newton scheme in some specific situations, we have not observed a general improvement of the algorithm over a broad range of problems.

To evaluate the proposed method, a number of different fluid problems are considered. All of these problems use a particular spatial discretization based on a pressure stabilized Petrov–Galerkin finite element formulation of the low Mach number Navier–Stokes equations with heat and mass transport. Computational results are presented for several challenging CFD benchmark problems as well as two large scale 3D flow simulations.

A major goal of this work is to study robustness and efficiency issues related to inexact Newton schemes and to explore the limits of effectiveness of these methods. There are alternate approaches to solving difficult nonlinear problems starting from

[3] For definitions of the various types of convergence referred to in this paper, see [3].

poor initial guesses, such as false time stepping and continuation schemes, but these are not considered here.

2. THE INEXACT NEWTON METHOD

We write the system of nonlinear algebraic equations that results from discretization of the fluid flow equations as

$$F(u) = 0, \quad F: R^n \rightarrow R^n. \tag{2.1}$$

We assume throughout that F is continuously differentiable and denote its Jacobian (matrix) by $F' \in R^{n\times n}$. Given an initial approximate solution u_0 of (2.1), classical Newton's method determines a sequence of approximate solutions by $u_{k+1} = u_k + s_k$, where the step s_k is characterized by the *Newton equation*

$$F'(u_k)s_k = -F(u_k). \tag{2.2}$$

In an inexact Newton method [2], the Newton equation is relaxed to an *inexact Newton condition*

$$\|F(u_k) + F'(u_k)s_k\| \leq \eta_k \|F(u_k)\| \tag{2.3}$$

for some $\eta_k \in [0, 1)$, where $\|\cdot\|$ is a norm of choice. This formulation naturally allows the use of an iterative linear algebra method: One first chooses η_k and then applies the iterative solver to (2.2) until an s_k is determined for which the residual norm satisfies (2.3). In this context, η_k is often called a *forcing term,* since its role is to force the residual of (2.2) to be suitably small.

Note that $F(u_k) + F'(u_k)s_k$ is both the residual of the (2.2) and the local linear model of $F(u_k + s_k)$ given by the first-order terms of the Taylor series of F at u_k. Thus in reducing the linear residual to satisfy (2.3), one will also make progress in reducing the nonlinear residual as long as there is sufficiently good agreement between F and its the local linear model at the step s_k.

It is shown in [2] that, near a solution of (2.1) at which F' is invertible, the local convergence of an inexact Newton method is controlled by the forcing terms. In particular, one can obtain local convergence that is as fast as desired, up to the (typically q-quadratic) convergence of Newton's method, by choosing the η_k's to be sufficiently small. For example, if $\eta_k \rightarrow 0$, then the convergence is typically q-superlinear, and if $\eta_k = O(\|F(u_k)\|)$, then the convergence is typically q-quadratic [2].

2.1. *The Backtracking Globalization*

To be practically effective, a Newton-like method requires globalization, i.e., augmentation with procedures that enhance the likelihood of convergence when u_0 is not near a solution. Various globalization procedures have been developed, primarily within the context of optimization (see, e.g., [3]), and some of these are discussed in Seciton 5. We focus primarily on *backtracking,* also known as *linesearch* or *damping,* in which steps are shortened as necessary until satisfactory steps are found. The specific backtracking algorithm that we consider here is the following form [4], which has also served as the basis of a general-purpose inexact Newton solver in [13].

ALGORITHM INB (Inexact Newton backtracking method [4]).

```
Let $u_0$, $\eta_{\max} \in [0, 1)$, $t \in (0, 1)$, and $0 < \theta_{\min} < \theta_{\max} < 1$ be given.
For $k = 0, 1, \ldots$ (until convergence) do:
    Choose initial $\eta_k \in [0, \eta_{\max}]$ and $s_k$ such that
            $\|F(u_k) + F'(u_k)s_k\| \leq \eta_k \|F(u_k)\|$.
    While $\|F(u_k + s_k)\| > [1 - t(1 - \eta_k)]\|F(u_k)\|$ do:
        Choose $\theta \in [\theta_{\min}, \theta_{\max}]$.
        Update $s_k \leftarrow \theta s_k$ and $\eta_k \leftarrow 1 - \theta(1 - \eta_k)$.
    Set $u_{k+1} = u_k + s_k$.
```

Note that, for a given initial η_k, a satisfactory initial s_k exists if the Newton equation (2.2) is consistent, in particular if $F'(u_k)$ is invertible. If a satisfactory initial s_k can be found, then we have from remarks in [4, Section 6] that Algorithm INB does not break down in the while-loop; i.e., an acceptable s_k is determined after at most a finite number of step reductions. Furthermore, it is easy to see that an inexact Newton condition (2.3) holds for each s_k and η_k determined in the while-loop and, in particular, for the final s_k and η_k. Thus each final step s_k determined by the algorithm satisfies both (2.3) and

$$\|F(u_k + s_k)\| \leq [1 - t(1 - \eta_k)]\|F(u_k)\|, \tag{2.4}$$

which can be viewed as a sufficient decrease condition on $\|F\|$. To shed light on this condition, we denote the *actual reduction* in $\|F\|$ by $\text{ared}_k \equiv \|F(u_k)\| - \|F(u_k + s_k)\|$ and define the *predicted reduction* given by the local linear model to the $\text{pred}_k \equiv \|F(u_k)\| - \|F(u_k) + F'(u_k)s_k\|$. Then (2.3) and (2.4) are equivalent, respectively, to $\text{pred}_k \geq (1 - \eta_k)\|F(u_k)\|$ and $\text{ared}_k \geq t(1 - \eta_k)\|F(u_k)\|$. In particular, if the inexact Newton condition (3) requires the predicted reduction to be at least $(1 - \eta_k)\|F(u_k)\|$, then the sufficient decrease condition (2.4) requires the actual reduction to be at least the fraction t of that amount.

Algorithm INB offers strong global convergence properties combined with potentially fast local convergence. We have the following theoretical result from [4].

THEOREM 2.1 [4]. *Assume that F is continuously differentiable. If $\{u_k\}$ produced by Algorithm* INB *has a limit point u_* such that $F'(u_*)$ is invertible, then $F(u_*) = 0$ and $u_k \rightarrow u_*$.* Furthermore, the initial s_k and η_k *are accepted without modification in the while-loop for all sufficiently large k.*

Note in particular that if the iteration sequence has any limit point at which F' is invertible, then that point must be a solution of (2.1) and the iterates must converge to it. Furthermore, the asymptotic convergence is governed by the initial η_k's as in the local convergence analysis of [2], and desirably fast convergence can be obtained by taking them to be suitably small.

2.2. *The Forcing Terms*

The forcing terms η_k not only determine the asymptotic speed of convergence to a solution of (2.1) but also affect the efficiency and robustness of the algorithm away from a solution. Indeed, away from a solution, choosing a very small η_k and solving (2.2) with corresponding accuracy may result in a step s_k so long that $F(u_k + s_k)$ disagrees significantly with the local linear model $F(u_k) + F'(u_k)s_k$, an outcome termed *oversolving* in [5]. Oversolving may result in little or no decrease in $\|F\|$ and, consequently, may necessitate backtracking to obtain an acceptable step; see Section 4.3.2 for an illustration. Even if an acceptable decrease in $\|F\|$ is obtained, it may be undesirably small relative to the expense of obtaining such an accurate solution of (2.2). A less accurate solution of (2.2), in addition to costing less, might give more reduction in $\|F\|$ and place less burden on the backtracking.

We have implemented two forcing term choices from [5] that tend to minimize oversolving while giving desirably fast asymptotic convergence to a solution of (1). These are as follows:

Choice 1. Select any $\eta_0 \in [0, 1)$ and choose

$$\eta_k = \frac{F(u_k)\| - \|F(u_{k-1}) + F'(u_{k-1})s_{k-1}\| \,|}{\|F(u_{k-1})\|}, \quad k = 1, 2, \ldots . \tag{2.5}$$

Choice 2. Given $\gamma \in [0, 1]$ and $\alpha \in (1, 2]$, select any $\eta_0 \in [0, 1)$ and choose

$$\eta_k = \gamma \left(\frac{\|F(u_k)\|}{\|F(u_{k-1})\|} \right)^{\alpha}, \quad k = 1, 2, \ldots . \tag{2.6}$$

In our implementation, we use the initial value $\eta_0 = 10^{-2}$ with the above choices. Also, to ensure that $\eta_k \leq \eta_{\max}$ in Algorithm INB, we follow (2.5) and (2.6) with the safeguard

$$\eta_k \leftarrow \min\{\eta_k, \eta_{\max}\}. \tag{2.7}$$

It is observed in [13] that local convergence results in [5, Theorems 2.2, 2.3] can be combined with Theorem 2.1 above to obtain the following convergence theorems for Algorithm INB when the initial η_k's are determined by (2.5) or (2.6) subject to (2.7).

THEOREM 2.2 [13]. *Assume that F is continuously differentiable and that each η_k in Algorithm* INB *is given by* (2.5) *followed by* (2.7). *If $\{u_k\}$ produced by Algorithm* INB *has a limit point u_* such that $F'(u_*)$ is invertible, then $F(u_*) = 0$ and $u_k \to u_*$. Furthermore, if F' is Lipschitz continuous at u_*, then*

$$\|u_{k+1} - u_*\| \leq \beta \|u_k - u_*\| \, \|u_{k-1} - u_*\|, \quad k = 1, 2, \ldots, \tag{2.8}$$

for a constant β independent of k.

Remark. As noted in [5], it follows from (2.8) that the convergence is q-superlinear, two-step q-quadratic, and of r-order $(1 + \sqrt{5})/2$.

THEOREM 2.3 [13]. *Assume that F is continuously differentiable and that each η_k in Algorithm* INB *is given by* (2.6) *followed by* (2.7). *If $\{u_k\}$ produced by Algorithm* INB *has a limit point u_* such that $F'(u_*)$ is invertible, then $F(u_*) = 0$ and $u_k \to u_*$. Furthermore, if F' is Lipschitz continuous at u_*, then the following hold: If $\gamma < 1$, then the convergence is of q-order α; if $\gamma = 1$, then the convergence is of r-order α and of q-order p for every $p \in [1, \alpha)$.*

In keeping with [5, 13], we also implement the following safeguards, which are applied after η_k has been determined by (2.5) or (2.6) and before applying (2.7):

Choice 1 *safeguard.* Modify η_k by $\eta_k \leftarrow \max\{\eta_k, \eta_{k-1}^{(1+\sqrt{5})/2}\}$ if $\eta_{k-1}^{(1+\sqrt{5})/2} > 0.1$.
Choice 2 *safeguard.* Modify η_k by $\eta_k \leftarrow \max\{\eta_k, \gamma\eta_{k-1}^{\alpha}\}$ if $\gamma\eta_{k-1}^{\alpha} > 0.1$.

The purpose of these is to prevent the initial η_k's from becoming too small far away from a solution. This can happen coincidentally with either (2.5) or (2.6); it can also happen with (2.5) when backtracking forces a very short step that results in very good agreement between F and its local linear model. Note that if $\{u_k\}$ converges to a solution of (2.1) at which F' is invertible and Lipshitz continuous, then we have $\eta_k \to 0$ with either (2.5) or (2.6). It follows that these safeguards eventually become inactive and do not affect the asymptotic convergence given in Theorems 2.2 and 2.3.

2.3. *Other Details of the Implementation*

Various remaining details of our implementation of Algorithm INB are as follows: We use $\eta_{\max} = 0.9$; this fairly large value allows the η_k's to become correspondingly large if necessary to reduce oversolving. We use $t = 10^{-4}$; this very small value results in accepting almost any step that gives a reduction in $\|F\|$. The use of $\theta_{\min}$ and $\theta_{\max}$ in Algorithm INB to determine minimal and maximal steplength reduction is known as *safeguarded backtracking* in the optimization community. In keeping with common practice, we use $\theta_{\max} = \frac{1}{2}$ and $\theta_{\min} = \frac{1}{10}$ and determine $\theta \in [\theta_{\min}, \theta_{\max}]$ to minimize a quadratic $p(t)$ that satisfies $p(0) = \frac{1}{2}\|F(u_k)\|^2$, $p(1) = \frac{1}{2}\|F(u_k + s_k)\|^2$, and $p'(0) = (d/dt)\frac{1}{2}\|F(u_k + ts_k)\|^2|_{t=0}$. The norm is a weighted Euclidean norm with weights that reflect problem scaling.

Successful termination is declared if $\|F(u_k)\| \leq \varepsilon_F\|F(u_0)\|$, where $\varepsilon_F = 10^{-2}$ in the experiments in Section 4 below, *and* the steplength criterion,

$$\frac{1}{n}\|Ws_k\|_2 < 1,$$

is also satisfied, where n is the total number of unknowns and W is a diagonal weighting matrix with entries

$$W_{ii} = \frac{1}{\varepsilon_r|u_k^{(i)}| + \varepsilon_a},$$

in which $u_k^{(i)}$ is the ith component of u_k and $\varepsilon_r = 10^{-3}$ and $\varepsilon_a = 10^{-8}$ in the experiments in Section 4. In our experience, this second criterion is typically more stringent and is necessary to ensure that finer physical details of the flow and transport are adequately resolved. Essentially, it requires that each variable of the Newton correction be small relative to its current value. This assures that all variables (even variables with small magnitude) are considered appropriately in determining when to halt the Newton iteration. This weight matrix definition is similar to a criterion used to dynamically control time step sizes and is standard in general purpose ODE packages such as LSODE [9].

3. THE DISCRETIZED EQUATIONS

The governing PDEs used in our experiments are given below. In these equations the unknown quantities are $\mathbf{u}$, P, T, and Y_k; these are, respectively, the fluid velocity vector, the hydrodynamic pressure, the temperature, and the mass fraction for species k:

Momentum transport:

$$\rho \mathbf{u} \cdot \nabla \mathbf{u} - \nabla \cdot \mathbf{T} - \rho \mathbf{g} = \mathbf{0}. \tag{3.1}$$

Total mass observation:

$$\nabla \cdot \mathbf{u} = \mathbf{0}. \tag{3.2}$$

Energy transport:

$$\rho C_p \mathbf{u} \cdot \nabla T + \nabla \cdot \mathbf{q} = \mathbf{0}. \tag{3.3}$$

Species mass transport:

$$\rho \mathbf{u} \cdot \nabla Y_k + \nabla \cdot \mathbf{j}_k = 0. \tag{3.4}$$

In these equations, ρ, $\mathbf{g}$, and C_p are, respectively, the density, the gravity vector, and the specific heat at constant pressure. The necessary constitutive equations for $\mathbf{T}$, $\mathbf{q}$, and $\mathbf{j}_k$ are given by (3.5)–(3.7) below:

Stress tensor:

$$\mathbf{T} = -P\mathbf{I} + \mu\{\nabla \mathbf{u} + \nabla \mathbf{u}^{\mathrm{T}}\}. \tag{3.5}$$

Heat flux:

$$\mathbf{q} = -\kappa \nabla T. \tag{3.6}$$

Species mass flux:

$$\mathbf{j}_k = -\rho D_k \nabla Y_k, \quad k = 1, ..., N-1. \tag{3.7}$$

Here μ, κ, and D_k are, respectively, the dynamic viscosity, the thermal conductivity, and the diffusion coefficient of species k in the mixture.

The above equations are derived by assuming a constant property, multicomponent dilute Newtonian fluid mixture with no chemical reactions. Additionally, the

Mach number is assumed low so that effects of viscous dissipation can be neglected in the energy transport equation (3.3). More information on this system of equations can be found in [19].

Finally, to complete the system, boundary conditions are imposed on (3.1)–(3.4) by taking combinations of Dirichlet conditions on $\mathbf{u}$, P, T, and Y_k and specified flux conditions on $\mathbf{T}$, q, and $\mathbf{j}_k$. In Section 4.2, we discuss the specific boundary conditions for each test problem in more detail.

To obtain an algebraic system of equations (2.1), a Petrov–Galerkin weighted residual formulation of (3.1)–(3.4) is used. This scheme utilizes a pressure stabilized Petrov–Galerkin (PSPG) formulation to allow equal-order interpolation of velocity and pressure, along with streamline upwinding (SUPG) to limit oscillations due to high grid Reynolds and Peclet numbers. This formulation follows the work of Hughes *et al.* [10] and Tezduyar [21]. Specifically, the discrete equations are obtained from the following equations:

Momentum:

$$\begin{aligned} F_{\mathbf{u}} = &\int_{\Omega} [\rho\mathbf{u}\cdot\nabla\mathbf{u} - \nabla\cdot\mathbf{T} - \rho\mathbf{g}]\Phi d\Omega \\ &+ \int_{\Omega^e} \tau_{\text{supg}}^{(\mathbf{u})} (\mathbf{u}\cdot\nabla\Phi)[\rho\mathbf{u}\cdot\nabla\mathbf{u} - \nabla\cdot\mathbf{T} - \rho\mathbf{g}]\, d\Omega. \end{aligned} \tag{3.8}$$

Total mass:

$$F_P = \int_{\Omega} [\nabla\cdot\mathbf{u}]\Phi d\Omega + \int_{\Omega^e} \tau_{pspg} \nabla\Phi\cdot[\mathbf{u}\cdot\nabla\mathbf{u} - \nabla\cdot\mathbf{T} - \rho\mathbf{g}]\, d\Omega. \tag{3.9}$$

Energy:

$$F_T = \int_{\Omega} [\rho C_p\mathbf{u}\cdot\nabla T + \nabla\cdot\mathbf{q}]\Phi d\Omega + \int_{\Omega^e} \tau_{\text{supg}}^{(T)} (\mathbf{u}\cdot\nabla\Phi)[\rho C_p\mathbf{u}\cdot\nabla T + \nabla\cdot\mathbf{q}]\, d\Omega. \tag{3.10}$$

Species mass:

$$F_{Y_k} = \int_{\Omega} [\rho\mathbf{u}\cdot\nabla Y_k + \nabla\cdot\mathbf{j}_k]\Phi d\Omega + \int_{\Omega^e} \tau_{\text{supg}}^{(Yk)} (\mathbf{u}\cdot\nabla\Phi)[\rho\mathbf{u}\cdot\nabla Y_k + \nabla\cdot\mathbf{j}_k]\, d\Omega. \tag{3.11}$$

In these, the stability parameters (the τ's) are functions of the fluid velocity, $\mathbf{u}$, and are given in [10, 21, 19].

To form the Jacobian F' of the system (3.1), we first linearize all terms of (3.8)–(3.11), except those containing the stability parameters. The discrete form of these linearized terms is determined by expanding the unknowns $\mathbf{u}$, P, T, and Y_k and the weighting function Φ in terms of a linear finite element basis. The contribution to F' resulting from these terms is then computed by analytic evaluation. Finally, the contribution to F' of the terms containing the stability parameters is computed by numerical differentiation and added to the analytically evaluated terms. The resulting Newton equation (2.2) is a fully coupled nonsymmetric linear system.

4. NUMERICAL EXPERIMENTS

4.1. *The Testing Environment*

The inexact Newton method outlined in Section 2 was tested by incorporating it in a parallel finite element reacting flow code called MPSalsa [15]. This code implements the Petrov–Galerkin formulation described in Section 3 in a distributed data setting through a process outlined briefly as follows: The underlying finite element grid is subdivided over processors using the graph partitioning package CHACO [8], so that the number of finite element nodes in each subdomain is balanced and the communication cost (essentially proportional to the surface area or perimeter of each subdomain) is minimized. Using this decomposition, MPSalsa sets up the finite element discretization, with each processor producing a subset of the discretized equations and Jacobian entries corresponding to its assigned subdomain. At each inexact Newton iteration, the Newton equation (2.2) is approximately solved using the parallel iterative solver package Aztec [11], which provides a number of solver and preconditioner options.

For this study, we restricted the iterative solver to the restarted GMRES method [14] and the preconditioner to a domain decomposition scheme using an incomplete factorization, ILU(0) [12, 20], within subdomains. For the two 3D problems described in Section 4.4, this preconditioner corresponds to extracting a block diagonal matrix from the original matrix (where each block is associated with the local unknowns on a particular processor) and producing an ILU(0) factorization of this matrix. For the three benchmark problems described in Section 4.2, the preconditioner is similar. However, each block or local processor based matrix is augmented by the set of equations associated with its neighboring unknowns (points updated by neighboring processors but connected by an edge in the finite element stencil connectivity graph to an unknown on this processor). Connections to unknowns outside of the processor's assigned unknowns or neighboring unknowns are discarded. Thus, equations appearing in one processor's matrix may also appear in another processor's matrix. An ILU(0) factorization is produced on each local augmented matrix and is essentially applied to each processor's assigned unknowns.[4] This procedure corresponds to overlapping the subdomain preconditioning matrices. Though this preconditioner requires additional storage compared to the unaugmented systems, it can significantly improve the overall convergence. More details on Aztec, GMRES, and these parallel preconditioners can be found in [20].

At each inexact Newton iteration, MPSalsa generates the Jacobian of the discretized system by a combination of analytic evaluation and numerical differentiation as described in Section 3 above. The Jacobian is then used in Aztec for the matrix–vector products required by GMRES. In the present context, these products are very computationally efficient, and this approach is considerably more economical for the test problems considered here than a "matrix free" approach in which these products are approximated by finite-differences of F-values. The ILU(0)

[4] The ILU preconditioner is applied to both assigned unknowns and neighbor unknowns. However, values from neighbor unknowns are discarded as different values for these unknowns are computed on neighboring processors.

preconditioner factors are computed from the new Jacobian at each inexact Newton step, and this computation entailed considerable expense in our tests. A strategy that would allow re-use of preconditioner factors over a number of inexact Newton steps might reduce this expense considerably, but we have not pursued such strategies in this study.

All tests reported here were run on Intel Paragons operated by Sandia National Laboratories.

4.2. *Three Standard Benchmark Problems*

The three test problems described below are standard benchmark problems used for verification of fluid flow codes and solution algorithms. In all cases, the GMRES restart value was 200, which was sufficiently large that GMRES stagnation did not become an issue for even the most difficult of the linear subproblems generated by the inexact Newton algorithm. We also allowed a maximum of 600 GMRES iterations at each inexact Newton step, after which the GMRES iterations were terminated and a new inexact Newton step started even if the condition (2.3) did not hold. In all cases, the initial approximate solution was the zero vector.

4.2.1. The Thermal Convection Problem. This standard benchmark problem [1] consists of the thermal convection (or buoyancy driven) flow of a fluid in a differentially heated square box in the presence of gravity. It requires the solution of the momentum transport, energy transport, and total mass conservation equations defined in Section 3 on the unit square in the plane with the following Dirichlet and Neumann boundary conditions:

$$T = T_{\text{cold}}, \quad u = v = 0 \text{ at } x = 0, \tag{4.1}$$

$$T = T_{\text{hot}}, \quad u = v = 0 \text{ at } x = 1, \tag{4.2}$$

$$\frac{\partial T}{\partial y} = 0, \quad u = v = 0 \text{ at } y = 0, \tag{4.3}$$

$$\frac{\partial T}{\partial y} = 0, \quad u = v = 0 \text{ at } y = 1. \tag{4.4}$$

When Eqs. (3.1)–(3.3) and the boundary conditions (4.1)–(4.4) are suitably nondimensionalized, two parameters appear, the Prandtl number Pr and the Rayleigh number Ra. In our study we took Pr $= 1$ and varied the magnitude of the Rayleigh number. As the magnitude of Ra is increased the nonlinear effects of the convection terms increase and the solution becomes more difficult to obtain. A typical solution for this problem is shown in Fig. 4.1. All solutions for this problem were computed on a 100×100 equally spaced mesh, which resulted in 40,624 unknowns for the discretized problems. Twenty Paragon processors were used for all runs.

4.2.2. The Lid Driven Cavity Problem. This is a standard benchmark problem [7, 17] consisting of a confined flow in a square box driven by a moving boundary on the upper wall. This problem requires the solution of Eqs. (3.1)–(3.2) defined in Section 3 on the unit square with the following Dirichlet boundary conditions:

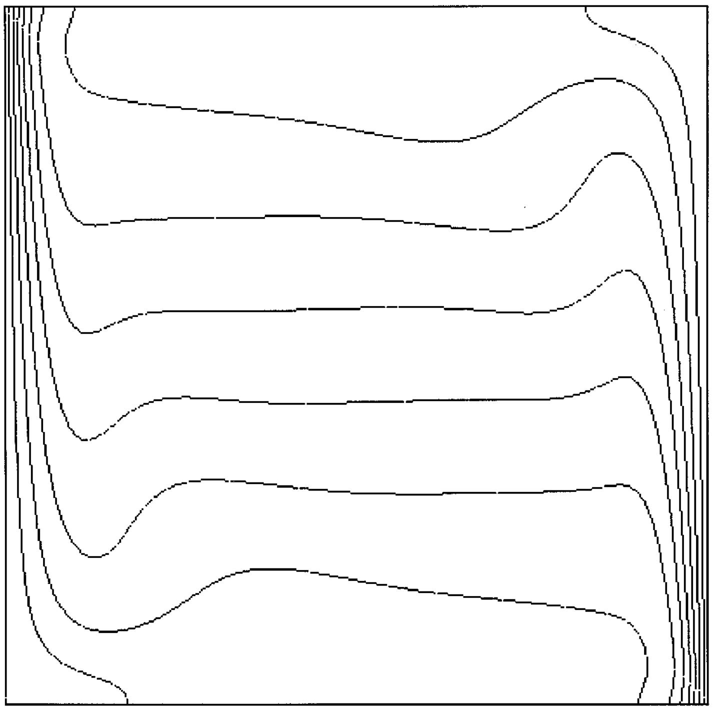

FIG. 4.1. Thermal convection in a square cavity at Ra = 1,000,000: Contour plot of temperature shows a thermal boundary layer along hot and cold walls.

$$u = v = 0 \quad \text{at } x = 0. \tag{4.5}$$

$$u = v = 0 \quad \text{at } x = 1. \tag{4.6}$$

$$u = v = 0 \quad \text{at } y = 0. \tag{4.7}$$

$$u = U_0, \quad v = 0 \quad \text{at } y = 1. \tag{4.8}$$

When Eqs. (3.1)–(3.2) and the boundary conditions (4.5)–(4.8) are suitably nondimensionalized, one parameter, the Reynolds number Re, appears. As Re is increased the nonlinear inertial terms in the momentum equation (1) become more dominant and the solution becomes more difficult to obtain. A typical solution for this problem is shown in Fig. 4.2. As in Section 4.2.1, all solutions were computed using a 100×100 equally spaced mesh, which resulted in 30,486 unknowns for the discretized problems. Twenty Paragon processors were used for all runs.

4.2.3. The Backward-Facing Step Problem. This is a standard benchmark problem [6] consisting of a rectangular channel with a 1×30 aspect ratio in which a reentrant backward-facing step is simulated by introducing a fully developed parabolic velocity profile in the upper half of the inlet boundary and imposing zero velocity on the lower half. As the fluid flows downstream it produces a recirculation zone on the lower channel wall, and for sufficiently high Re it also produces a recirculation zone farther downstream on the upper wall. This test problem requires

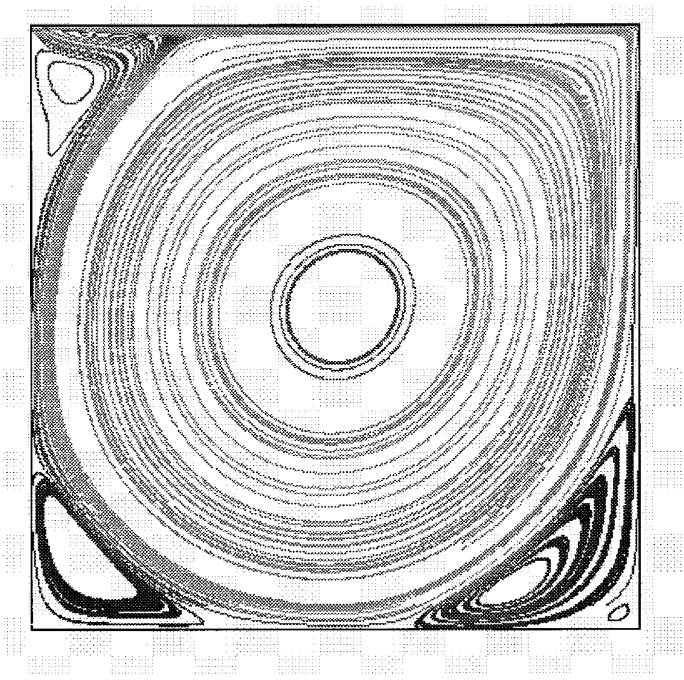

FIG. 4.2. Lid driven cavity at Re = 10,000: Contour plot of the stream function shows a main vortex and existence of corner vortices.

the solution of the same nondimensional equations as in the lid driven cavity problem. The boundary conditions are given by

$$u(y) = 24y(0.5 - y), v = 0 \quad \text{at } x = 0, 0 \leq y \leq 0.5, \tag{4.9}$$

$$u = v = 0 \quad \text{at } x = 0, -0.5 \leq y < 0, \tag{4.10}$$

$$\mathbf{T}_{xx} = 0, \mathbf{T}_{xy} = 0 \quad \text{at } x = 30, \tag{4.11}$$

$$u = v = 0 \quad \text{at } y = -0.5, \tag{4.12}$$

$$u = v = 0 \quad \text{at } y = 0.5. \tag{4.13}$$

As Re is increased the nonlinear inertial terms in the momentum equation (3.1) become more dominant and the solution becomes more difficult to obtain. A typical solution for this problem is shown in Fig. 4.3. All solutions for this problem were computed on a 20 × 400 unequally spaced mesh (not shown), which resulted in

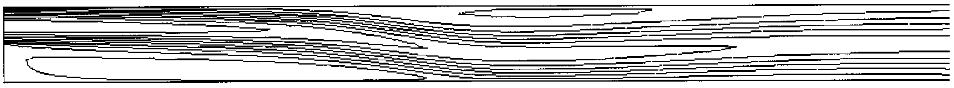

FIG. 4.3. Backward-facing step solution at Re = 800: Contour plot of the x-velocity shows recirculation on lower and upper walls.

25,623 unknowns for the discretized problems. Sixteen Paragon processors were used for all runs.

4.3. *Experiments with the Benchmark Problems*

In this section we report on experiments aimed at showing the effects of backtracking and of various choices of the forcing terms η_k in (2.3). We have used the problems introduced in Section 4.2 for these experiments because they are well-understood benchmark problems. Illustrative results for these problems are shown in the tables and figures in this section. The full set of test results for these problems is given in the Appendix.

4.3.1. *An illustration of backtracking.* Backtracking or other forms of globalization are often omitted in engineering codes. While an unglobalized inexact Newton method can be effective in special situations, e.g., when the initial guess is close to the final solution or the problem is almost linear, it will often fail to converge in more general circumstances. To illustrate the effects of backtracking, we show in Fig. 4.4 convergence histories with and without backtracking over the first 200 GMRES iterations (spanning a number of inexact Newton steps) for the backward-facing step problem with Reynolds number 600. In particular, denoting the approximate solution of the Newton equation at each GMRES iterations by $\tilde{s}$, we plot log $\|F(u_k + \tilde{s})\|$, where u_k is the approximate solution of the nonlinear system at the current Newton step. The solid curve shows log $\|F(u_k + \tilde{s})\|$ when backtracking is

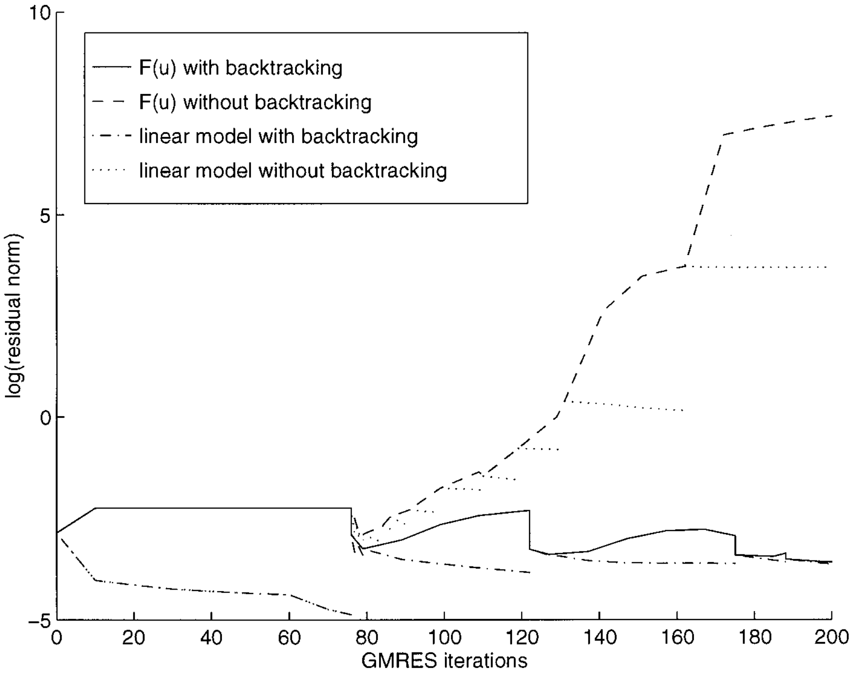

FIG. 4.4. Convergence history, first 200 GMRES iterations.

enabled while the dashed curve shows these values when backtracking is not used. The vertical intervals in the solid curve indicate the occurrence of backtracking. Note that these curves are identical at the first inexact Newton step through the first 76 GMRES iterations (the apparent plateau is actually a period of very slow increase in $\log \|F(u_k + \bar{s})\|$); however, they diverge at the end of that step as a result of backtracking and continue to diverge increasingly thereafter. Safeguarded Choice 1 forcing terms were used, with $\eta_0 = 10^{-2}$; this fairly small η_0 accounts for the relatively large number of GMRES iterations at the first step. The dotted and dash-dotted curves in Fig. 4.4 correspond to the linear model norm, i.e., $\log \|F(u_k) + F'(u_k)\bar{s}\|$, for the nonbacktracking and backtracking cases, respectively. Note that there is considerable divergence of $\log \|F(u_k + \bar{s})\|$ and $\log \|F(u_k) + F'(u_k)\bar{s}\|$ at each inexact Newton step, both with and without backtracking. Thus the safeguarded Choice 1 forcing terms failed to maintain good agreement between the nonlinear residual and the local linear model during the first 200 GMRES iterations, and backtracking was necessary to ensure a decrease in the nonlinear residual norm. For perspective, we show in Fig. 4.5 the entire convergence history for the backtracking case. From this, one sees that, after the first 200 GMRES iterations, backtracking was necessary during occasional periods of difficulty, but eventually good agreement was maintained between the nonlinear residual and the local linear model and convergence was ultimately obtained without further backtracking.

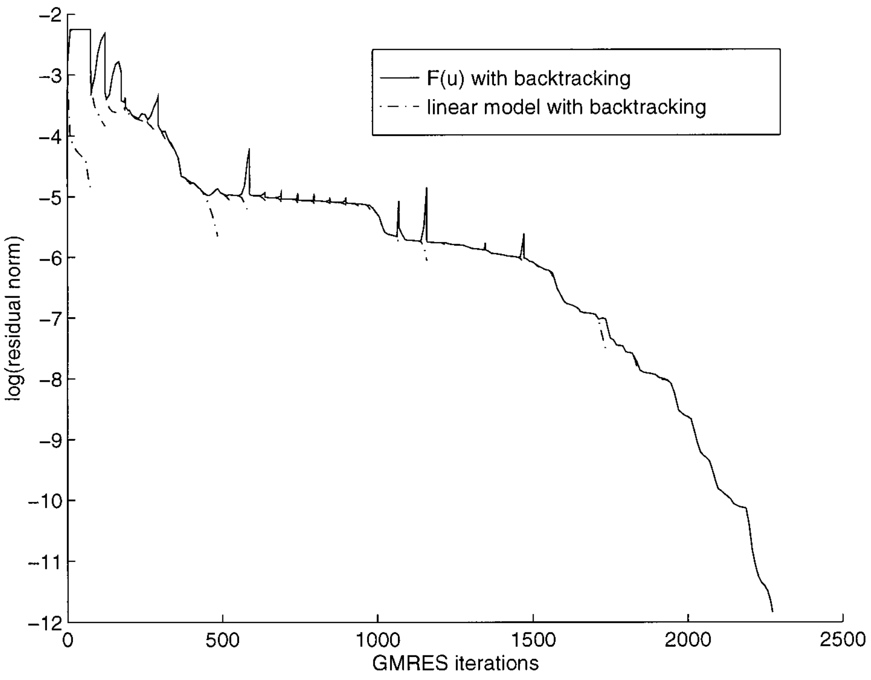

FIG. 4.5. Entire convergence history.

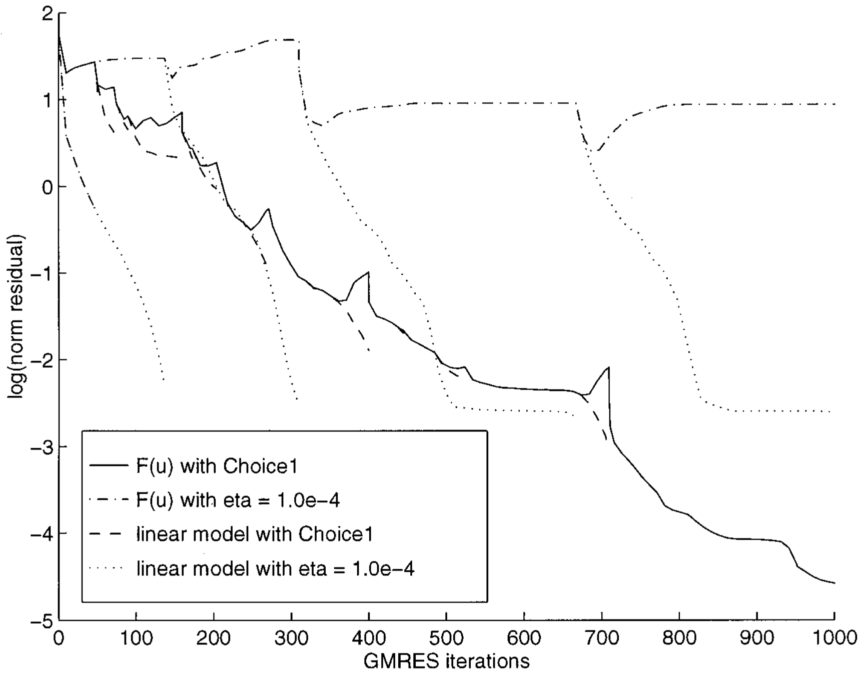

FIG. 4.6. Convergence history, first 1000 GMRES iterations.

4.3.2. *An illustration of oversolving.* To illustrate the issue of oversolving, we show in Figs. 4.6 and 4.7 convergence histories for two different forcing term choices, Choice 1 and $\eta_k = 10^{-4}$, both with backtracking, for the lid driven cavity problem at Reynolds number 1000. For each forcing term choice, the plots show the nonlinear residual norms and linear model norms versus the number of GMRES iterations as in the previous figures. Oversolving is indicated by the gaps between these curves, in which GRMES continues to reduce the linear model norm while the nonlinear residual norm typically stagnates or even increases. Oversolving is associated with significant disagreement between the linear model and the nonlinear residual; once it begins, subsequent GMRES iterations are usually wasted effort and may even be counterproductive. With the Choice 1 forcing terms, modest oversolving is evident until just beyond 700 GMRES iterations but is subsequently too small to be visible in the plots. With $\eta_k = 10^{-4}$, oversolving is much more pronounced and continues for many more GMRES iterations; convergence is ultimately obtained but much less efficiently than with the Choice 1 forcing terms.

4.3.3. *A robustness study.* We conducted a comprehensive study involving the benchmark problems with the goal of assessing the general robustness of an inexact Newton method with and without backtracking and with various choices of the forcing terms. In this study, the parameters that determine the difficulty of the benchmark problems were varied over wide ranges.

The forcing term choices included in the study were safeguarded Choices 1 and 2 and two constant choices. For Choice 2, we used $\gamma = 0.9$ and allowed $\alpha = 1.5$,

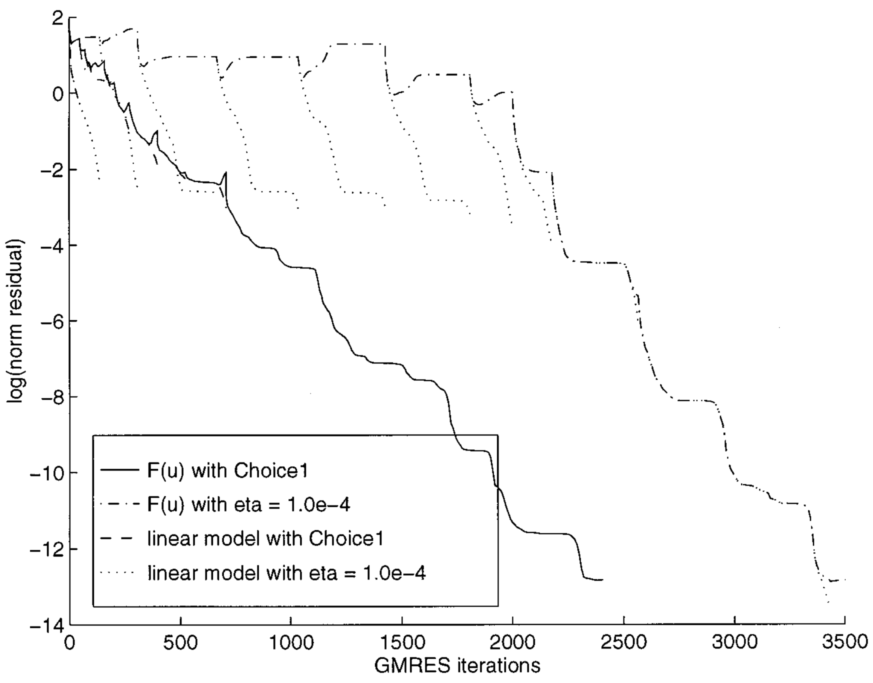

FIG. 4.7. Entire convergence history.

which gives asymptotic convergence of q-order 1.5 (about the same as the r-order $(1 + \sqrt{5})/2$ convergence of Choice 1), and $\alpha = 2$, which gives asymptotic q-quadratic convergence. The two constant choices have occasionally been used by others and represent somewhat different approaches. The first, $\eta_k = 10^{-1}$, requires only moderate accuracy in solving the Newton equation (2) at each inexact Newton step and gives only moderately fast asymptotic q-linear convergence near a solution. The second, $\eta_k = 10^{-4}$, requires considerable accuracy and results in a step that should usually give about the same performance as the exact Newton step, with very fast asymptotic q-linear convergence near a solution.

The results of the study are shown in Table I, which shows numbers of failures for these forcing term choices with and without backtracking. To help show where failures occurred, we have somewhat arbitrarily divided cases into "easier" and "harder" parameter ranges for each problem. (However, some of the "easier" problems may not be easy in any absolute sense.) For the thermal convection problem, the "easier" problems are with Ra = 10^3, 10^4, and 10^5; the "harder" problem is with Ra = 10^6. For the lid driven cavity problem, the "easier" problems are with Re = 1,000, 2,000, 3,000, 4,000, and 5,000; the "harder" problems are with Re = 6,000, 7,000, 8,000, 9,000, and 10,000. For the backward facing step problem, the "easier" problems are with Re = 100, 200, 300, 400, and 500; the "harder" problems are with Re = 600, 700, 750, and 800.

Table I shows that backtracking generally improves robustness for every choice of the forcing terms considered here. Indeed, in only one case above ($\eta_k = 10^{-1}$, backward facing step, "harder" problems) did backtracking result in more failures than no backtracking; we have no explanation for this case and regard it as anomalous. Table I also shows that the Choice 1 and 2 forcing terms generally give greater robustness than the constant choices, with or without backtracking. However, the table also shows that neither backtracking nor an effective forcing term choice alone is sufficient; both are necessary for good robustness. The best combination seen in Table I is Choice 1 with backtracking, followed closely by Choice 2, $\alpha = 2$, with backtracking.

4.3.4. *An efficiency comparison of Choice 1 and Choice 2 forcing terms.* We follow the robustness study above with a study aimed at assessing the relative efficiency of the Choice 1 and Choice 2 forcing terms on the benchmark problems when backtracking is used. The constant forcing term choices used above are not included here because their high failure rates precluded obtaining a sufficiently broad set of test problems. However, we note that in cases in which these constant choices succeeded, they often resulted in much less efficiency than the Choice 1 or Choice 2 forcing terms; see, in particular, the results in the Appendix for the backward facing step and lid driven cavity problems, in which the constant choice $\eta_k = 10^{-4}$ is notably less efficient than the Choice 1 or Choice 2 forcing terms.

As in Section 4.3.3 above, in Choice 2, we took $\gamma = 0.9$ and used $\alpha = 1.5$ and $\alpha = 2$ in this study. The test cases considered were those in which all three of these forcing term choices resulted in success, as follows: Ra $= 10^3$, 10^4, 10^5, and 10^6 for the thermal convection problem; Re $=$ 1,000, 3,000, 4,000, 5,000, 7,000, 8,000, 9,000, and 10,000 for the lid driven cavity problem; and Re $=$ 100, 200, 300, 400, and 500 for the backward facing step problem.

The results of the study are shown in Table II, which, for the different forcing

TABLE I
Distribution of Failures

	Thermal convection		Lid driven cavity		Backward-facing step	
Forcing term η_k	Easier	Harder	Easier	Harder	Easier	Harder
Choice 1	0	0	0	0	0	1
	0	1	0	5	0	4
Choice 2	0	0	1	1	0	3
$\alpha = 1.5$	0	1	1	4	0	4
Choice 2	0	0	0	0	0	2
$\alpha = 2$	0	1	1	5	1	4
10^{-1}	0	0	4	5	1	4
	0	1	5	5	1	2
10^{-4}	0	0	3	4	2	4
	0	1	5	5	3	4

Note. For each choice of η_k, the upper and lower lines are the number of failures with and without backtracking, respectively.

TABLE II
Forcing Term Comparison

Forcing term η_k	Inexact Newton steps	Backtracks	GMRES iterations	Time (seconds)
Choice 1	36.5	41.4	4054.1	792.1
Choice 2, $\alpha = 1.5$	36.3	49.8	4189.6	824.2
Choice 2, $\alpha = 2$	32.8	48.5	3951.6	779.4

Note. "Backtracks" gives arithmetic means; all other columns give geometric means.

term choices, gives mean numbers of inexact Newton steps, backtracks, and GMRES iterations, and mean run times (in seconds). All are geometric means except in the case of backtracks, in which they are arithmetic means. (There were no backtracks in some cases, and so geometric means were not defined for backtracks.)

Overall, Choice 1 and Choice 2, $\alpha = 2$, performed slightly better than Choice 2, $\alpha = 1.5$, which finished last in every category except inexact Newton steps, in which it essentially tied Choice 1. In comparing Choice 1 to Choice 2, $\alpha = 2$, it is notable that the former required fewer backtracks while the latter required fewer inexact Newton steps. This is not surprising: Choice 1 is aimed directly at maintaining good agreement between the nonlinear residual and the local linear model and, consequently, should relieve the backtracking of much of its burden; Choice 2, $\alpha = 2$, is more "aggressive" and gives asymptotic q-quadratic convergence, which may result in more backtracking away from the solution but reduce the number of inexact Newton steps in the end.

We also carried out a similar comparison involving only Choice 1 and Choice 2, $\alpha = 2$, on a somewhat larger test set on which both of these choices gave success. The results are similar to those in Table II, and so they are not included here.

4.4. *Experiments with Two 3D Problems*

In Sections 4.2 and 4.3, we have shown the effects on method performance of backtracking and various forcing term choices through illustrative examples and statistical studies involving three well-known 2D benchmark problems. These studies show, in particular, that backtracking coupled with an effective forcing term choice can lead to very significant overall improvement in robustness and efficiency over a range of problems. However, in our testing, we also observed considerable variations in the performance of different method options on individual problems; it was by no means true that a particular set of options always worked best.

In this section, in order to illustrate variations in method behavior as well as to show performance on particular realistic problems, we outline specific case studies of two large-scale 3D flow simulations.

4.4.1. A CVD Reactor Transport Problem. This example problem involves computing the 3D solution for fluid flow, heat transfer, and the mass transfer of three chemical species in a horizontal tilted chemical vapor deposition (CVD)

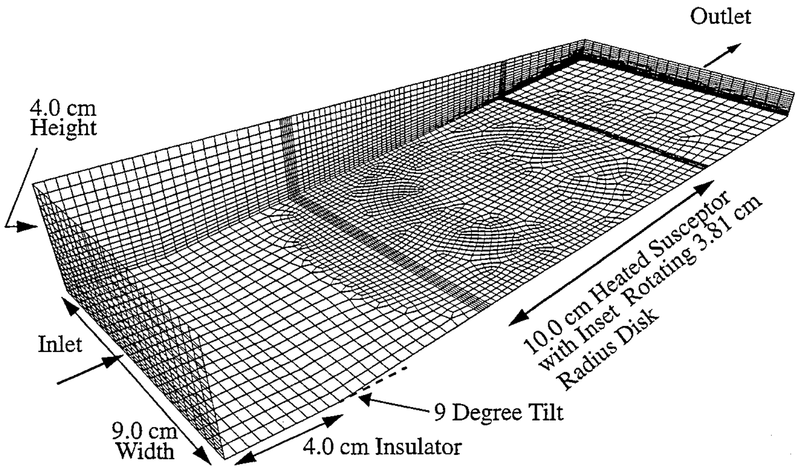

FIG. 4.8. Unstructured finite element mesh of CVD reactor.

reactor. The problem has three fluid velocities, the hydrodynamic pressure, temperature, and three chemical species as unknowns at each finite element node. The CVD reactor has a rectangular cross section with a tilted lower surface with an embedded spinning disk which cannot be accurately represented with a structured mesh (see Fig. 4.8). Fluid enters in the larger cross sectional area inlet and accelerates up the inclined surface with the inset rotating heated disk. At the elevated disk temperature, chemical reactions are initiated to deposit gallium arsenide (GaAs). In this example, we only transport the precursors for this reaction (tri-methylgallium,

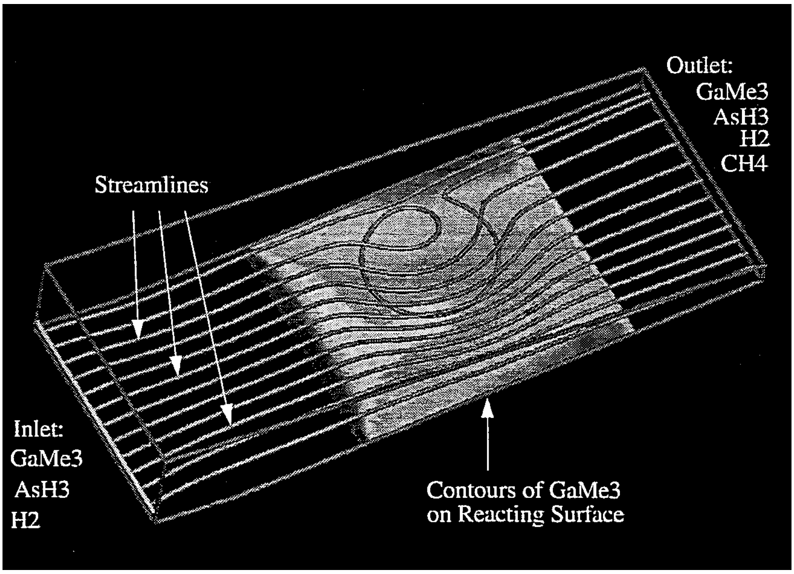

FIG. 4.9. Flow streamlines and contour plot of $GaMe_3$ mass fraction in CVD reactor.

TABLE III
Results for the CVD Reactor Problem with Backtracking

Forcing term η_k	Inexact Newton steps	Backtracks	GMRES iterations	Time (seconds)
Choice 1	25	3	1503	924.9
10^{-1}	13	1	1315	593.8
10^{-4}	5	0	1531	444.5

$GaMe_3$, arsine, AsH_3) and a carrier gas (hydrogen, H_2) and do not allow chemical reactions. In our example calculation, the inlet velocity is 60 cm/s, the inlet temperature is 600 K, and the disk rotates at 200 rpm and is heated to 900 K. To simulate the deposition process, we use a Dirichlet condition on the reactants that introduces significant diffusion gradients and boundary layers that approximate the average behavior of the full reacting system depositing GaAs on the rotating disk. (Results for the full reacting CVD system can be found in [18, 16]). In practice CVD reactors are run at low pressures and fluid velocities, and thus the Reynolds numbers are small ($\mathrm{Re} \approx 1.0$). Therefore, SUPG stabilization was not needed. In addition, for gasses at these temperatures and pressures, the Prandtl number and the Schmidt number (analogous to the Prandtl number) for mass transport are approximately one as well. A typical flow solution is shown in Fig. 4.9, where the streamlines show the effect of the counterclockwise rotation of the disk. Included is a contour plot of the concentration of tri-methylgallium at the heated surface. This contour plot is from the full reacting flow solution [16].

For these experiments, the number of unknowns for the discretized problem was 384,200. The number of Paragon processors used was 220. The GMRES restart value was 100, with a maximum of 600 GMRES iterations allowed at each inexact Newton step. Since these experiments are intended to be illustrative, we considered only three representative forcing term choices, viz., Choice 1 and the two constant choices, $\eta_k = 10^{-1}$ and $\eta_k = 10^{-4}$. Results for these forcing term choices, with and without backtracking, are shown in Tables III and IV. It is notable that, for this problem, performance was worse with backtracking than without for every forcing term choice. Furthermore, the best performances (at least in terms of time) were

TABLE IV
Results for the CVD Reactor Problem without Backtracking

Forcing term η_k	Inexact Newton steps	Backtracks	GMRES iterations	Time (seconds)
Choice 1	2	0	1052	707.9
10^{-1}	12	0	1051	511.5
10^{-4}	5	0	1531	445.5

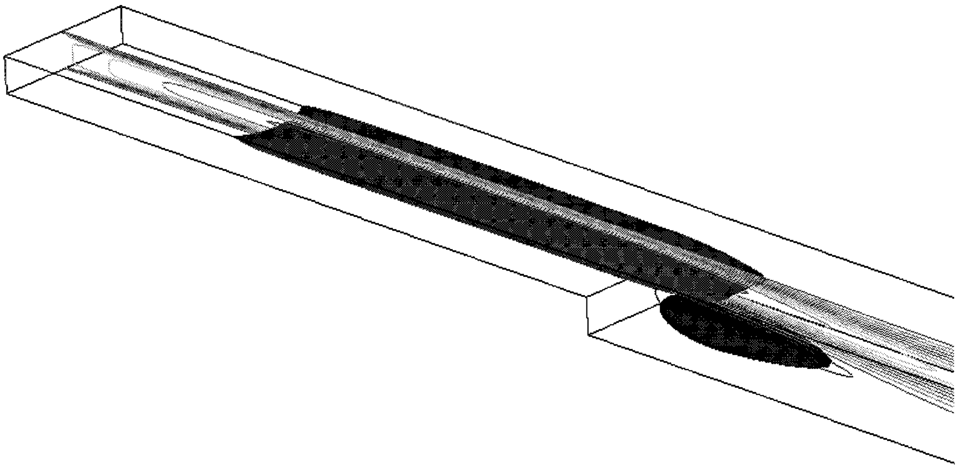

FIG. 4.10. The duct flow problem (top view) showing a center-plane plot of x-velocity contours. Isosurfaces show an accelerated flow region on the upper portion of the duct and a lower recirculation region.

from the constant choices without backtracking. In fact, the choice $\eta_k = 10^{-4}$, which was the clear winner in terms of time, took far fewer inexact Newton steps than the other two choices and never invoked backtracking even when it was allowed; i.e., initial inexact Newton steps were always acceptable.

4.4.2. A 3D Duct Flow Problem with Contamination Transport. This 3D problem models the steady flow of air through an expanding cross section duct. On the lower surface of the duct a recirculation region forms as in the backward-facing step example. The physical problem of interest is to solve for the flow field and for the downstream transport of ionized air molecules produced from a small source of nuclear ionizing radiation located on the lower wall behind the step. The goal of the simulation is to computationally predict the presence of the nuclear contamination in concentrations high enough to detect experimentally by special sensors. This information is to be used to aid in the economical decommissioning of old nuclear facilities.

The numerical computation requires the solution of the momentum transport, total mass, and contaminant species conservation equations defined in Section 3. The domain is a duct of width 0.2 m by 8 m long, with an inlet height of 0.1 m and outlet of 0.2 m. The mesh is finer near the solid walls and the step expansion location. The Reynolds number based on the outlet height is 400 and the Schmidt number is 1.0. A typical solution for this problem is shown in Fig. 4.10. This figure shows a contour plot of the x-velocity on the center-plane of the duct, an isosurface plot of an accelerated flow region in the upper duct, and a lower 3D recirculation region with negative velocities located behind the step. For these experiments, the number of unknowns for the discretized problem was 477,855. The GMRES restart value was 160, and a maximum of 640 GMRES iterations was allowed at each

TABLE V
Results for the Duct flow Problem

Forcing term η_k	Inexact Newton steps	Backtracks	GMRES iterations	Time (seconds)
Choice 1	28	6	13,450	3554.5
10^{-1}	25	7	15,477	3953.9
10^{-4}	24	6	15,360	3915.1

inexact Newton step. The number of Paragon processors used was 256. The forcing terms considered here are Choice 1 and the two constant choices $\eta_k = 10^{-1}$ and $\eta_k = 10^{-4}$.

Backtracking was necessary for success on this problem; the method diverged without backtracking. Results with backtracking are given in Table V. Notice that while Choice 1 required the most inexact Newton steps, it also took the fewest number of GMRES iterations and, consequently, the least amount of CPU time, despite the time required to create new Jacobians and preconditioners at the additional inexact Newton steps. We should note that the run-time advantage of Choice 1 over $\eta_k = 10^{-4}$ would have been even larger had the maximum number of GMRES steps been set higher. Indeed, the linear solver frequently took the maximum number of steps (640) without achieving convergence when $\eta_k = 10^{-4}$, and thus a larger maximum number of GMRES steps would have resulted in an even greater difference between the Choice 1 run times and the $\eta_k = 10^{-4}$ run times.

In the true physical problem of interest here, there is also a volumetric ion reaction source term. Results for these solutions are very similar to those of Table V. In a later manuscript, we will consider the inclusion of the reaction terms in the transport equations and study the convergence of the inexact Newton methods.

5. A COMMENT ON TRUST REGION METHODS

In addition testing not reported here, we also experimented with several variations of the backtracking algorithm that employed techniques associated with trust region methods. In a trust region method, steps are constrained to lie within spherical or ellipsoidal neighborhoods in which the linear model is “trusted” to represent the nonlinear residual well. Within each such neighborhood, a step is chosen to minimize approximately the norm of the local linear model; the size of the neighborhood is then adjusted for the next step to reflect agreement of the local linear model and the nonlinear residual. A popular trust region implementation is the *dogleg method,* in which a step is chosen to minimize the local linear model norm within the trust region along the *dogleg curve,* which joins the current point, the steepest descent point (the minimizer of the local linear model norm in the steepest descent direction), and the point determined by the Newton step. We omit further details and refer the reader to [3].

We carried out a number of experiments with a straightforward extension of the dogleg schema to the inexact Newton context, in which the inexact Newton step played the role of the Newton step.[5] In these tests, we experimented with several different norms (to properly capture the different physical scales of the variables) in defining trust region neighborhoods. For the most part our results indicate that, in the present context, such a trust-region method can be fairly competitive with backtracking. However, we noticed no paticular improvement in overall robustness, although occasionally one method converged when another would not and vice versa.[6] Moreover, the trust region method usually required slightly more CPU time than the backtracking method: There were fewer function evaluations due to less backtracking, but this savings was outweighed by the cost of more inexact Newton steps due to the scaling back of steps.

In addition to this dogleg implementation, we also conducted some experiments involving the following: (1) imposing a trust region-type steplength constraint on initial inexact Newton steps used in backtracking; (2) backtracking from initial inexact Newton steps along the dogleg curve, rather than simply scaling back the steps as in straightforward backtracking; (3) modifying GMRES so that the first iterate is the steepest descent step.[7] In no case did we observe any overall advantage over the straightforward backtracking method.

6. SUMMARY DISCUSSION

We have proposed an inexact Newton method with a backtracking globalization for the solution of the steady transport equations for momentum, heat, and mass transfer in flowing fluids. The algorithm offers choices of the forcing terms (the criteria that determine the accuracy of solutions of the linear subproblems) that are intended to enhance the robustness and efficiency of the method by maintaining good agreement between the nonlinear residual and its local linear model at each inexact Newton step. Theoretical support for the algorithm shows that it has strong global convergence properties together with desirably fast (up to q-quadratic) local convergence.

Extensive testing on three standard 2D benchmark problems has shown that both backtracking and an effective forcing term choice can greatly improve robustness. However, neither alone is sufficient; both are necessary for the best overall perfor-

[5] The MPSalsa testing environment was modified to allow the computation of products of the transpose of the Jacobian with vectors, which in turn allowed the computation of steepest descent steps.

[6] Only with the constant choice $\eta_k = 10^{-1}$ did we observe that the trust region method was more robust than backtracking. However, usually the CPU times for this choice were longer than for either Choice 1 or Choice 2 with backtracking.

[7] This simple modification requires an initial product $F'(x_k)^{\mathrm{T}}F(x_k)$, followed by an additional dot product and "saxpy" at each iteration.

mance. In our tests on the benchmark problems, the greatest overall robustness was obtained with backtracking in combination with the Choice 1 forcing terms, followed closely by the Choice 2 forcing terms with $\alpha = 2$. (See Section 2.2 for these forcing term formulations). Compared to Choice 1, Choice 2 with $\alpha = 2$ tended to require more backtracks but to take fewer inexact Newton steps in our experiments.

Tests on two 3D problems have shown the effectiveness of the algorithm on realistic large-scale flow simulations. Results for the second problem, a duct flow problem, reflect the overall results on the 2D benchmark problems: Backtracking was necessary for success, and, with backtracking, Choice 1 forcing terms resulted in considerably greater efficiency than either of two constant choices considered. In contrast, for the first problem, a CVD reactor problem, all three of these choices gave better performance *without* backtracking, and the best performance was obtained with the very small constant choice $\eta_k = 10^{-4}$. This demonstrates that no single strategy is best for all problems; the best course is to have a number of options available.

APPENDIX

On the following pages, we give the full set of test results for the experiments on the benchmark problems described in Sections 4.2, 4.3. For each problem, the first column of results gives values of the appropriate parameter, viz., the Rayleigh number Ra for the backward-facing step and thermal convection problems and the Reynolds number Re for the lid driven cavity problem. The second column (S/F) shows a success/failure flag: "0" indicates success; "-1" indicates either failure to succeed within the maximum allowable number of inexact Newton steps (either 100 or 200 steps in each case) or stagnation, as determined by failure to achieve sufficient reduction in the nonlinear residual norm for fifteen consecutive steps; "-2" indicates backtracking failure, i.e., failure to determine a successful step within the maximum allowable number of backtracks (eight). The third through eighth columns show, respectively, numbers of inexact Newton steps (Newt), numbers of function evaluations (#f( )), numbers of backtracks (Bkt), numbers of GMRES iterations (GMRES), final residual norms ($\|r\|$), and total run times in seconds (time). In some cases, test runs were terminated because of exceeding the allowable run time, machine failure, or other reasons. In these cases, if ultimate failure was clear, the runs were not repeated; these runs are indicated by `''terminated but clearly failing''` in the tables.

ACKNOWLEDGMENTS

The authors would like to thank Andrew Salinger for the use of the 3D CVD mesh and for helpful suggestions and debugging related to the MPSalsa analytic Jacobian entries.

Backward Facing Step problem without backtracking

Choice 1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	12	24	0	696	6.926e-11	1.355e+02
Re_200	0	12	24	0	741	5.316e-10	1.449e+02
Re_300	0	15	30	0	925	4.657e-10	1.779e+02
Re_400	0	27	54	0	1123	2.430e-10	2.461e+02
Re_500	0	40	80	0	1107	2.002e-07	3.033e+02
Re_600	-1	200	400	0	56120	2.950e+04	9.096e+03
Re_700	-1	200	400	0	57134	8.542e+03	9.215e+03
Re_750	-1	200	400	0	56041	1.586e+04	9.070e+03
Re_800	-1	200	400	0	65592	1.379e+04	1.055e+04

Choice 2 forcing terms, gamma = .9, alpha = 1.5

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	11	22	0	695	3.951e-10	1.308e+02
Re_200	0	13	26	0	816	2.906e-11	1.582e+02
Re_300	0	16	32	0	970	3.715e-10	1.845e+02
Re_400	0	24	48	0	1054	3.460e-09	2.249e+02
Re_500	0	44	88	0	1436	1.776e-08	3.498e+02
Re_600	-1	200	400	0	60995	2.378e+03	9.850e+03
Re_700	-1	200	400	0	57442	1.145e+04	9.308e+03
Re_750	-1	100	200	0	28814	3.524e+03	4.675e+03
Re_800	-1	100	200	0	28805	2.772e+03	4.671e+03

Choice 2 forcing terms, gamma = .9, alpha = 2

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	10	20	0	732	1.566e-12	1.375e+02
Re_200	0	11	22	0	733	1.135e-11	1.470e+02
Re_300	0	14	28	0	817	4.458e-11	1.718e+02
Re_400	0	24	48	0	1115	1.316e-10	2.419e+02
Re_500	-1	200	400	0	56808	1.652e+04	9.320e+03
Re_600	-1	127	...	(terminated but clearly failing)			
Re_700	-1	100	200	0	28936	1.772e+03	4.777e+03
Re_750	-1	100	200	0	26319	3.997e+03	4.296e+03
Re_800	-1	100	200	0	26044	4.213e+03	4.217e+03

Eta = 1.0e-4 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	6	12	0	906	3.063e-10	1.501e+02
Re_200	0	8	16	0	1378	2.367e-10	2.363e+02
Re_300	-1	200	400	0	84412	2.362e+04	1.363e+04
Re_400	-1	100	200	0	38874	7.057e+03	6.325e+03
Re_500	-1	100	200	0	39932	1.660e+03	6.389e+03
Re_600	-1	100	200	0	35472	3.013e+03	5.689e+03
Re_700	-1	100	200	0	36867	1.702e+03	5.892e+03
Re_750	-1	100	200	0	41772	1.467e+03	6.678e+03
Re_800	-1	100	200	0	25773	4.706e+03	4.160e+03

Eta = 1.0e-1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	9	18	0	657	4.448e-08	1.138e+02
Re_200	0	11	22	0	887	6.528e-09	1.488e+02
Re_300	0	11	22	0	871	7.445e-08	1.462e+02
Re_400	0	13	26	0	1024	5.929e-08	1.759e+02
Re_500	-1	200	400	0	61845	1.293e+04	1.003e+04
Re_600	-1	134	...	(terminated but clearly failing)			
Re_700	0	23	46	0	1405	2.939e-07	2.496e+02
Re_750	0	30	60	0	2098	2.820e-08	3.658e+02
Re_800	-1	100	200	0	36931	2.014e+03	5.974e+03

Backward Facing Step problem with backtracking

Choice 1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	10	21	1	799	6.673e-12	1.376e+02
Re_200	0	14	32	4	937	2.651e-11	1.755e+02
Re_300	0	17	41	7	1024	3.046e-10	2.059e+02
Re_400	0	36	89	17	1440	6.626e-10	3.252e+02
Re_500	0	57	166	52	2045	5.616e-06	4.838e+02
Re_600	0	90	293	113	3709	2.504e-06	8.307e+02
Re_700	0	135	500	230	6889	1.911e-08	1.435e+03
Re_750	-1	50	284	184	1511	1.231e-01	5.024e+02
Re_800	0	146	484	192	8766	2.572e-08	1.667e+03

Choice 2 forcing terms, gamma = .9, alpha = 1.5

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	10	21	1	794	2.087e-10	1.403e+02
Re_200	0	15	34	4	951	2.392e-10	1.880e+02
Re_300	0	16	38	6	1049	9.707e-11	2.150e+02
Re_400	0	35	87	17	1412	2.965e-09	3.262e+02
Re_500	0	65	197	67	2556	4.443e-09	6.099e+02
Re_600	-1	37	228	154	1658	9.957e-02	4.370e+02
Re_700	-1	42	257	173	1322	1.353e-01	4.383e+02
Re_750	0	125	440	190	6591	3.441e-08	1.339e+03
Re_800	-1	31	182	120	666	1.742e-01	2.911e+02

Choice 2 forcing terms, gamma = .9, alpha = 2

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	9	19	1	799	9.126e-14	1.422e+02
Re_200	0	13	31	5	882	7.090e-12	1.840e+02
Re_300	0	14	35	7	1009	1.456e-11	1.972e+02
Re_400	0	30	76	16	1290	2.562e-10	2.974e+02
Re_500	0	49	136	38	1947	7.029e-09	4.555e+02
Re_600	0	86	283	111	3687	7.603e-08	8.777e+02
Re_700	0	139	593	315	8160	1.814e-08	1.660e+03
Re_750	-1	40	215	135	1203	1.346e-01	3.905e+02
Re_800	-1	43	237	151	1601	1.106e-01	4.528e+02

Eta = 1.0e-4 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	7	15	1	1064	2.751e-11	1.802e+02
Re_200	0	9	22	4	1600	6.582e-11	2.763e+02
Re_300	0	11	30	8	2357	7.338e-11	4.031e+02
Re_400	-2	23	146	101	6419	1.180e-01	1.111e+03
Re_500	-2	7	47	34	1850	3.836e-01	3.255e+02
Re_600	-2	17	101	68	5619	1.437e-01	9.338e+02
Re_700	-2	14	92	65	4402	2.939e-01	7.430e+02
Re_750	-2	29	207	150	12090	2.309e-01	2.016e+03
Re_800	-2	34	282	215	18797	3.417e-01	3.151e+03

Eta = 1.0e-1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_100	0	10	21	1	900	8.528e-09	1.459e+02
Re_200	0	12	29	5	988	2.199e-08	1.680e+02
Re_300	0	13	35	9	1173	9.388e-08	2.000e+02
Re_400	-2	13	101	76	1194	2.819e-01	2.510e+02
Re_500	0	19	55	17	676	1.542e-07	2.854e+02
Re_600	-1	67	571	437	10987	1.592e-01	2.129e+03
Re_700	-2	12	92	69	1022	4.108e-01	2.179e+02
Re_750	-2	39	330	253	5086	3.271e-01	1.017e+03
Re_800	-1	25	215	165	2274	4.127e-01	4.831e+02

Lid Driven Cavity problem without backtracking

Choice 1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	0	30	60	0	2122	2.630e-15	4.541e+02
Re_2000	0	41	82	0	3565	5.205e-15	7.446e+02
Re_3000	0	51	102	0	6164	8.029e-12	1.208e+03
Re_4000	0	71	142	0	5647	3.389e-12	1.219e+03
Re_5000	0	83	166	0	6527	1.844e-11	1.417e+03
Re_6000	-1	200	400	0	67183	6.207e+03	1.174e+04
Re_7000	-1	200	400	0	81214	1.002e+03	1.411e+04
Re_8000	-1	200	400	0	74175	5.448e+03	1.289e+04
Re_9000	-1	200	400	0	86770	3.853e+03	1.500e+04
Re_10000	-1	131	...	(terminated but clearly failing)			

Choice 2 forcing terms, gamma = .9, alpha = 1.5

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	0	23	46	0	2565	2.270e-15	5.116e+02
Re_2000	0	34	68	0	3347	3.583e-14	6.811e+02
Re_3000	0	56	112	0	5248	5.928e-12	1.081e+03
Re_4000	-1	200	400	0	78050	3.680e+03	1.357e+04
Re_5000	0	83	166	0	5366	3.389e-11	1.239e+03
Re_6000	0	115	230	0	7170	1.072e-12	1.687e+03
Re_7000	-1	200	400	0	84788	1.215e+03	1.467e+04
Re_8000	-1	100	200	0	36641	5.696e+03	6.427e+03
Re_9000	-1	100	200	0	38108	2.879e+03	7.307e+03
Re_10000	-1	100	200	0	32570	7.362e+03	5.744e+03

Choice 2 forcing terms, gamma = .9, alpha = 2

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	0	40	80	0	2662	2.968e-15	6.010e+02
Re_2000	0	29	58	0	3509	1.181e-12	6.907e+02
Re_3000	0	47	94	0	5577	7.942e-12	1.113e+03
Re_4000	0	61	122	0	6750	8.215e-12	1.375e+03
Re_5000	-1	200	400	0	76219	2.659e+03	1.330e+04
Re_6000	-1	200	400	0	80877	8.966e+03	1.404e+04
Re_7000	-1	100	200	0	31671	1.940e+03	5.634e+03
Re_8000	-1	100	200	0	40775	2.628e+03	7.736e+03
Re_9000	-1	100	200	0	35977	4.576e+03	6.265e+03
Re_10000	-1	50	100	0	18857	2.963e+03	3.266e+03

Eta = 1.0e-4 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	-1	200	400	0	96406	9.679e+03	1.662e+04
Re_2000	-1	80	...	(terminated but clearly failing)			
Re_3000	-1	100	200	0	49811	2.827e+03	9.376e+03
Re_4000	-1	100	200	0	43776	1.725e+04	7.565e+03
Re_5000	-1	50	100	0	25476	5.325e+03	4.374e+03
Re_6000	-1	50	100	0	24458	5.393e+03	4.194e+03
Re_7000	-1	50	100	0	25907	3.471e+03	4.457e+03
Re_8000	-1	50	100	0	22716	5.028e+03	3.908e+03
Re_9000	-1	50	100	0	26342	6.895e+03	4.519e+03
Re_10000	-1	50	100	0	26485	2.946e+03	4.551e+03

Eta = 1.0e-1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	-1	189	...	(terminated but clearly failing)			
Re_2000	-1	84	...	(terminated but clearly failing)			
Re_3000	-1	100	200	0	50882	2.334e+03	8.753e+03
Re_4000	-1	100	200	0	47878	3.394e+03	8.248e+03
Re_5000	-1	100	200	0	42786	6.101e+03	7.388e+03
Re_6000	-1	100	200	0	54918	3.909e+03	9.428e+03
Re_7000	-1	100	200	0	53013	4.337e+03	9.121e+03
Re_8000	-1	100	200	0	47922	5.469e+03	8.227e+03
Re_9000	-1	100	200	0	50902	2.043e+03	8.741e+03
Re_10000	-1	100	200	0	42473	8.496e+04	7.319e+03

Lid Driven Cavity problem with backtracking

Choice 1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	0	21	45	3	2790	2.567e-15	5.152e+02
Re_2000	0	36	87	15	4275	6.235e-14	8.062e+02
Re_3000	0	57	145	31	6987	1.630e-13	1.299e+03
Re_4000	0	69	174	36	7484	4.426e-12	1.426e+03
Re_5000	0	74	193	45	8124	2.479e-11	1.523e+03
Re_6000	0	86	232	60	9864	5.201e-12	1.836e+03
Re_7000	0	102	289	85	11148	1.207e-11	2.109e+03
Re_8000	0	120	330	90	16110	1.355e-11	2.977e+03
Re_9000	0	149	432	134	22112	2.249e-11	4.099e+03
Re_10000	0	183	547	181	26519	1.306e-10	4.886e+03

Choice 2 forcing terms, gamma = .9, alpha = 1.5

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	0	21	43	1	2613	2.594e-15	4.996e+02
Re_2000	-1	60	252	132	3757	4.030e-02	7.916e+02
Re_3000	0	51	135	33	6941	7.310e-13	1.287e+03
Re_4000	0	66	171	39	8410	2.624e-12	1.569e+03
Re_5000	0	70	188	48	7717	1.479e-11	1.451e+03
Re_6000	-1	37	152	78	2500	1.503e-01	5.144e+02
Re_7000	0	98	292	96	11195	6.132e-11	2.095e+03
Re_8000	0	125	389	139	16777	3.047e-11	3.109e+03
Re_9000	0	144	442	154	17824	9.513e-11	3.337e+03
Re_10000	0	185	599	229	26288	7.010e-11	4.861e+03

Choice 2 forcing terms, gamma = .9, alpha = 2

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	0	20	44	4	2826	2.509e-15	5.395e+02
Re_2000	0	34	88	20	4195	5.246e-15	8.062e+02
Re_3000	0	50	136	36	5952	2.237e-12	1.139e+03
Re_4000	0	62	184	60	7836	1.013e-11	1.478e+03
Re_5000	0	70	199	59	7326	1.525e-10	1.409e+03
Re_6000	0	85	251	81	9601	8.992e-12	1.847e+03
Re_7000	0	102	314	110	11214	5.275e-11	2.171e+03
Re_8000	0	116	341	109	15326	2.773e-11	2.844e+03
Re_9000	0	139	421	143	19517	6.589e-11	3.635e+03
Re_10000	0	180	580	220	27252	9.041e-11	5.011e+03

Eta = 1.0e-4 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	0	23	84	38	10267	2.865e-15	1.802e+03
Re_2000	0	21	75	33	10140	1.309e-14	1.769e+03
Re_3000	-2	26	133	82	14715	1.836e-01	2.589e+03
Re_4000	-1	22	122	78	11201	6.310e-01	1.977e+03
Re_5000	-2	42	211	128	24085	1.110e-01	4.241e+03
Re_6000	-1	25	134	84	14294	5.188e-01	2.521e+03
Re_7000	-2	11	58	37	5061	7.978e-01	9.010e+02
Re_8000	-2	6	32	21	2912	8.558e-01	5.153e+02
Re_9000	-1	33	161	95	18953	5.298e-01	3.336e+03
Re_10000	0	30	74	14	17138	3.374e-11	2.962e+03

Eta = 1.0e-1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Re_1000	0	22	51	7	2798	3.840e-14	5.230e+02
Re_2000	-2	30	146	87	3331	9.722e-02	6.415e+02
Re_3000	-2	27	144	91	3725	2.845e-01	7.173e+02
Re_4000	-1	36	181	109	4927	1.794e-01	9.482e+02
Re_5000	-1	26	128	76	3273	7.602e-02	6.321e+02
Re_6000	-2	27	126	73	3221	4.894e-02	6.114e+02
Re_7000	-2	14	63	36	1900	7.791e-02	3.630e+02
Re_8000	-2	18	79	44	2068	8.373e-02	3.925e+02
Re_9000	-1	42	226	142	6247	3.365e-02	1.237e+03
Re_10000	-2	22	100	57	2889	1.450e-01	5.576e+02

```
Thermal Convection in the Square problem without backtracking


Choice 1 forcing terms

              S/F   Newt   #f()  Bkt   GMRES      ||r||          time
Ra_1.0e03      0      6     12    0     2255   8.867e-15     4.303e+02
Ra_1.0e04      0     15     30    0     3370   1.786e-10     6.811e+02
Ra_1.0e05      0     29     58    0     4667   2.258e-11     1.011e+03
Ra_1.0e06     -1    200    400    0    64430   8.015e+03     1.281e+04


Choice 2 forcing terms, gamma = .9, alpha = 1.5

              S/F   Newt   #f()  Bkt   GMRES      ||r||          time
Ra_1.0e03      0      6     12    0     2482   3.385e-13     4.740e+02
Ra_1.0e04      0     11     22    0     3855   2.848e-12     7.460e+02
Ra_1.0e05      0     30     60    0     4265   1.031e-10     9.367e+02
Ra_1.0e06     -1    200    400    0    59388   4.135e+03     1.185e+04


Choice 2 forcing terms, gamma = .9, alpha = 2

              S/F   Newt   #f()  Bkt   GMRES      ||r||          time
Ra_1.0e03      0      5     10    0     2151   7.304e-13     4.146e+02
Ra_1.0e04      0      9     18    0     3506   2.677e-12     6.777e+02
Ra_1.0e05      0     20     40    0     3651   4.434e-11     7.630e+02
Ra_1.0e06     -1    200    400    0    57746   6.610e+03     1.156e+04


Eta = 1.0e-4 forcing terms

              S/F   Newt   #f()  Bkt   GMRES      ||r||          time
Ra_1.0e03      0      5     10    0     2101   7.060e-13     4.009e+02
Ra_1.0e04      0      8     16    0     4004   3.651e-13     7.750e+02
Ra_1.0e05      0     12     24    0     6619   2.909e-12     1.258e+03
Ra_1.0e06     -1    137  ... (terminated but clearly failing)


Eta = 1.0e-1 forcing terms

              S/F   Newt   #f()  Bkt   GMRES      ||r||          time
Ra_1.0e03      0      9     18    0     1877   2.518e-10     3.693e+02
Ra_1.0e04      0     12     24    0     3033   6.074e-11     5.753e+02
Ra_1.0e05      0     16     32    0     5154   2.747e-11     1.005e+03
Ra_1.0e06     -1    156  ... (terminated but clearly failing)
```

Thermal Convection in the Square problem with backtracking

Choice 1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Ra_1.0e03	0	6	12	0	2255	8.867e-15	4.261e+02
Ra_1.0e04	0	12	25	1	3399	3.050e-11	6.632e+02
Ra_1.0e05	0	22	52	8	3804	8.665e-13	7.937e+02
Ra_1.0e06	0	23	54	8	2628	2.263e-10	5.931e+02

Choice 2 forcing terms, gamma = .9, alpha = 1.5

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Ra_1.0e03	0	6	12	0	2482	3.385e-13	4.730e+02
Ra_1.0e04	0	11	22	0	3855	2.848e-12	7.477e+02
Ra_1.0e05	0	21	47	5	4561	4.043e-11	9.340e+02
Ra_1.0e06	0	27	61	7	2921	8.741e-11	6.485e+02

Choice 2 forcing terms, gamma = .9, alpha = 2

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Ra_1.0e03	0	5	10	0	2151	7.304e-13	4.130e+02
Ra_1.0e04	0	9	18	0	3506	2.677e-12	6.756e+02
Ra_1.0e05	0	18	42	6	3925	6.098e-12	7.894e+02
Ra_1.0e06	0	24	59	11	2993	8.966e-12	6.526e+02

Eta = 1.0e-4 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Ra_1.0e03	0	5	10	0	2101	7.060e-13	4.012e+02
Ra_1.0e04	0	8	16	0	4004	3.651e-13	7.729e+02
Ra_1.0e05	0	11	23	1	5830	1.919e-12	1.126e+03
Ra_1.0e06	0	22	83	39	10856	3.414e-11	2.143e+03

Eta = 1.0e-1 forcing terms

	S/F	Newt	#f()	Bkt	GMRES	||r||	time
Ra_1.0e03	0	9	18	0	1877	2.518e-10	3.671e+02
Ra_1.0e04	0	12	24	0	3033	6.074e-11	5.711e+02
Ra_1.0e05	0	15	37	7	3267	7.188e-11	6.502e+02
Ra_1.0e06	0	16	38	6	2366	4.164e-10	4.946e+02

REFERENCES

1. G. De Vahl Davis and C. P. Jones, Natural convection in a square cavity: A comparison exercise, *Int. J. Numer. Methods Fluids* **3** (1983).
2. R. S. Dembo, S. C. Eisenstat, and T. Steihaug, Inexact Newton methods, *SIAM J. Numer. Anal.* **19,** 400 (1982).
3. J. E. Dennis Jr. and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations,* Prentice-Hall, Englewood Cliffs, NJ, 1983. [Series in Automatic Computation]
4. S. C. Eisenstat and H. F. Walker, Globally convergent inexact Newton methods, *SIAM J. Optimization* **4,** 393 (1994).
5. S. C. Eisenstat and H. F. Walker, Choosing the forcing terms in an inexact Newton method, *SIAM J. Sci. Comput.* **17,** 16 (1996).
6. D. K. Gartling, A test problem for outflow boundary conditions—flow over a backward facing step, *Int. J. Numer. Methods Fluids* **11,** 953 (1990).
7. U. Ghia, K. N. Ghia, and C. T. Shin, High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method, *J. Comput. Phys.* **48,** 387 (1982).
8. B. Hendrickson and R. Leland, *The Chaco User's Guide—Version* 1.0, Tech. Rep. Sand93-2339, Sandia National Laboratories, Albuquerque NM, 87185, August 1993.
9. A. C. Hindmarsh, LSODE and LSODEI: Two new initial value ordinary differential equation solvers, *ACM Signum Newsletter* **15,** 10 (1980).
10. T. R. Hughes, L. P. Franca, and M. Balestra, A new finite element formulation for computational fluid dynamics: V. circumventing the Babuska-Brezzi condition: A stable Petrov–Galerkin formulation of the Stokes problem accommodating equal-order interpolations, *Comp. Meth. App. Mech. and Eng.* **59,** 85 (1986).
11. S. A. Hutchinson, J. N. Shadid, and R. S. Tuminaro, *Aztec user's guide—version 1.0,* Tech. Rep. Sand95-1559, Sandia National Laboratories, Albuquerque NM, 87185, Oct. 1995.
12. J. A. Meijerink and H. A. Van der Vorst, An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix, *Math. Comput.* **31,** 148 (1977).
13. M. Pernice and H. F. Walker, NITSOL: *a Newton Iterative Solver for Nonlinear Systems,* Tech. Rep. 5/96/85, Mathematics and Statistics Department, Utah State University, May 1996.
14. Y. Saad and M. H. Schultz, GMRES: a generalized minimal residual method for solving nonsymmetric linear systems, *SIAM J. Sci. Stat. Comput.* **7,** 856 (1986).
15. A. G. Salinger, K. D. Devine, G. L. Hennigan, H. K. Moffat, S. A. Hutchinson, and J. N. Shadid, **MPSalsa,** *A Finite Element Computer Program for Reacting Flow Problems.* Part 2—User's Guide, Tech. Rep. SAND96-2331, Sandia National Laboratories, Albuquerque, NM, 1996.
16. A. G. Salinger, J. N. Shadid, H. K. Moffat, S. A. Hutchinson, G. L. Hennigan, and K. D. Devine, *Massively parallel computation of 3d flow and chemistry in cvd reactors, Crystal Growth,* to be submitted.
17. R. Schreiber and H. B. Keller, Driven cavity flows by efficient numerical techniques, *J. Comput. Phys.* **49,** 310 (1983).
18. J. N. Shadid, S. A. Hutchinson, G. L. Hennigan, H. K. Moffat, K. D. Devine, and A. G. Salinger, Efficient parallel computation of unstructured finite element reacting flow solutions, *Parallel Comput.,* to appear.
19. J. N. Shadid, H. K. Moffat, S. A. Hutchinson, G. L. Hennigan, K. D. Devine, and A. G. Salinger, *MPSalsa: A Finite Element Computer Program for Reacting Flow Problems. Part 1: THEORETICAL DEVELOPMENT,* Tech. Rep. SAND96-2752, Sandia National Laboratories, May 1996.
20. J. N. Shadid and R. S. Tuminaro, Aztec—*A Parallel Preconditioned Krylov Solver Library: Algorithm Description Version* 1.0, Tech. Rep. Sand95: in preparation, Sandia National Laboratories, Albuquerque NM, 87185, August 1995.
21. T. E. Tezduyar, Stabilized finite element formulations for incompressible flow computations, *Adv. App. Mech.* **28,** 1 (1992).