SIAM J. SCI. STAT. COMPUT.
Vol. 11, No. 3, pp. 450–481, May 1990

003

HYBRID KRYLOV METHODS FOR NONLINEAR SYSTEMS OF EQUATIONS*

PETER N. BROWN† AND YOUCEF SAAD‡

Abstract. Several implementations of Newton-like iteration schemes based on Krylov subspace projection methods for solving nonlinear equations are considered. The simplest such class of methods is Newton's algorithm in which a (linear) Krylov method is used to solve the Jacobian system approximately. A method in this class is referred to as a Newton–Krylov algorithm. To improve the global convergence properties of these basic algorithms, hybrid methods based on Powell's dogleg strategy are proposed, as well as linesearch backtracking procedures. The main advantage of the class of methods considered in this paper is that the Jacobian matrix is never needed explicitly.

Key words. nonlinear systems, Krylov methods, inexact Newton methods, conjugate gradient techniques

AMS(MOS) subject classification. 65H10

1. Introduction.

We consider here several implementations of Newton-like iteration schemes for solving nonlinear systems of equations that we will refer to as *nonlinear Krylov subspace projection methods.* All these methods are based upon the idea of using a basic Newton iteration in which the Newton equations are solved approximately by an available Krylov method. The particular Krylov methods we will consider are *Arnoldi's Method* [22], and the *Generalized Minimum Residual Method* (GMRES) [24]. The Krylov methods have the virtue of requiring almost no matrix storage, resulting in a distinct advantage over direct methods for solving the Newton equations.

To be more specific, consider the nonlinear system of equations

$$F(u) = 0, \tag{1.1}$$

where F is a nonlinear function from $\mathbf{R}^N$ to $\mathbf{R}^N$. Newton's method applied to (1.1) results in the following iteration:

1. Set $u_0 =$ an initial guess.
2. For $n = 0, 1, 2, \cdots$ until convergence do:

$$\begin{aligned} &\text{Solve } J(u_n)\delta_n = -F(u_n), \\ &\text{Set } u_{n+1} = u_n + \delta_n, \end{aligned} \tag{1.2}$$

where $J(u_n) = F'(u_n)$ is the system Jacobian. For large problems, iterative methods are frequently used to solve (1.2) only approximately, giving rise to methods that

* Received by the editors November 20, 1987; accepted for publication (in revised form) March 23, 1989. This work was performed under the auspices of the U.S Department of Energy by the Lawrence Livermore National Laboratory under contract W-7405-Eng-48, and supported by the Department of Energy Office of Energy Research, Applied Mathematical Sciences Research Program.

† Computing and Mathematics Research Division, L-316, Lawrence Livermore National Laboratory, Livermore, California 94550. The work of this author was supported in part by National Science Foundation grant DMS-8506651.

‡ Research Institute for Advanced Computer Science, MS230-5, NASA Ames Research Center, Moffett Field, California 94035. This work was performed while this author was at the Center for Supercomputer Research and Development, University of Illinois, Urbana, Illinois 61801. The work of this author was supported in part by National Science Foundation grants DCR84-10110 and DCR85-09970, by U.S. Department of Energy grant DE-FG02-85ER25001, by U.S. Air Force contract AFSOR-85-0211, and from International Business Machines.

can be viewed as *inexact-Newton* methods [6]. We will refer to a Newton iteration in which a Krylov method is used to solve (1.2) approximately as a *nonlinear Krylov method.*

Typically, a Krylov method for solving (1.2) requires only the action of the Jacobian matrix J times a vector v, and not J explicitly. In the nonlinear equations setting, this action can be approximated by a difference quotient of the form

$$J(u)v \approx \frac{F(u+\sigma v) - F(u)}{\sigma}, \tag{1.3}$$

where u is the current approximation to a root of (1.1) and σ is a scalar. In [2], Brown has given an analysis of the resulting Newton/Krylov algorithms when (1.3) is used to approximate Jv, and has referred to them as *inexact-Newton/finite-difference projection methods.* Sufficient conditions are given in [2] on the size of the σ's in the finite-difference algorithms that guarantee the local convergence of the iteration. Here, we will be concerned with modifications of the above algorithms that are intended to guarantee the global convergence of the iteration to a local solution of (1.1). We recall that Newton's method converges only when the initial guess is close enough to a solution, so a modification is needed to guarantee convergence for arbitrary initial guesses. We refer to a method that converges for any initial guess as *globally convergent*, as opposed to a locally convergent method such as the unmodified Newton iteration. The first modification will add a linesearch backtrack procedure to the basic algorithm, whereas the second will incorporate a local quadratic model of the function $f(u) = \frac{1}{2}F(u)^T F(u)$ and will be a model trust region type algorithm.

We note that several authors have considered Krylov methods for solving the Newton equations approximately inside a Newton algorithm in the context of systems of ordinary differential equations [3]–[5],[11]. Also, Steihaug [25] and O'Leary [20] have used the *Conjugate Gradient* method in the unconstrained optimization of a real-valued function of several variables. Wigton, Yu, and Young [27] and more recently Kerkhoven and Saad [16] have accelerated nonlinear fixed point iterations of the form $u_{n+1} = M(u_n)$ by applying this approach to solving the nonlinear system of equations $u - M(u) = 0$. Note that, as was observed by Chan and Jackson [5], the new system of equations $u - M(u) = 0$ can be viewed as a nonlinearly preconditioned version of the original system of equations. This constitutes one way of preconditioning a nonlinear system and we should stress that it retains the nice feature of not requiring explicit Jacobians. A second approach for preconditioning a nonlinear system of equations is to exploit the fact that often the system separates naturally into a simple linear part, e.g., the Laplacian operator in partial differential equations, and the nonlinear part. In many cases the linear part may constitute a good preconditioner to the whole system, and fast direct solvers can be exploited to apply these preconditioners (e.g., see [1] and [26]). A more standard way of preconditioning a nonlinear system is to use the usual incomplete factorization techniques such as the incomplete LU (ILU) factorization. However, this approach requires the Jacobian matrix explicitly and thus loses the main advantage of Jacobian-free Krylov subspace methods. Nevertheless, we can easily imagine a procedure where the Jacobian is computed only occasionally, i.e., much less frequently than with a standard Newton approach, in order to derive a preconditioning.

In §2, we review the basic Krylov methods under consideration, and then in §3 we present the linesearch backtracking modification of the nonlinear Krylov iteration. In §4, we present a model trust region algorithm in connection with the nonlinear GMRES method, in §5 we discuss scaling and preconditioning of the linear and nonlinear

iterations, and then in §6 we present some numerical results on the above algorithms. Finally, in §7 we make some concluding remarks.

2. Nonlinear Krylov algorithms. In this section we will review the Krylov subspace methods under consideration, and discuss their main properties. We start with a brief description of the Arnoldi and GMRES algorithms, and then present their nonlinear versions, which combine them with a Newton iteration. Next, we comment on some implementation details of Arnoldi and GMRES, and then briefly discuss finite-difference versions and incomplete versions of the two methods.

Once again, we are interested in using a Newton-like iteration scheme to solve the nonlinear system

$$F(u) = 0, \tag{2.1}$$

where F is a nonlinear function from $\mathbf{R}^N$ to $\mathbf{R}^N$. As discussed in the Introduction, at each iteration we must obtain an approximate solution of the linear system (1.2), which we rewrite as

$$J\delta = -F, \tag{2.2}$$

where F and its Jacobian J are evaluated at the current iterate. If $\delta^{(0)}$ is an initial guess for the true solution of (2.2), then letting $\delta = \delta^{(0)} + z$, we have the equivalent system

$$Jz = r^{(0)}, \tag{2.3}$$

where $r^{(0)} = -F - J\delta^{(0)}$ is the initial residual. Let K_m be the *Krylov subspace*

$$K_m \equiv \text{span}\{r^{(0)}, Jr^{(0)}, \cdots, J^{m-1}r^{(0)}\}.$$

Arnoldi's method and GMRES both find an approximate solution

$$\delta^{(m)} = \delta^{(0)} + z^{(m)}\ , \text{with } z^{(m)} \in K_m, \tag{2.4}$$

such that either

$$(-F - J\delta^{(m)}) \perp K_m \text{ (equivalently } (r^{(0)} - Jz^{(m)}) \perp K_m) \tag{2.5}$$

for Arnoldi's method, or

$$\|F + J\delta^{(m)}\|_2 = \min_{\delta \in \delta^{(0)} + K_m} \|F + J\delta\|_2 \ \ (= \min_{z \in K_m} \|r^{(0)} - Jz\|_2) \tag{2.6}$$

for GMRES. Here, $\|\cdot\|_2$ denotes the Euclidean norm on $\mathbf{R}^N$ and orthogonality is meant in the usual Euclidean sense.

The following algorithm is a nonlinear version of the Arnoldi (GMRES) algorithm, which at every outer iteration generates an orthonormal system of vectors v_i $(i = 1, 2, \cdots, m)$ of the subspace K_m and then builds the vector $\delta^{(m)}$ that satisfies (2.5) (or (2.6) for GMRES). In both algorithms, v_1 is obtained by normalizing $r^{(0)}$.

Algorithm: Newton–Arnoldi (Newton–GMRES)

(1) *Start:* Choose u_0 and compute $F(u_0)$. Set $n = 0$. Choose a tolerance ϵ_0.
(2) *Arnoldi process:*

- For an initial guess $\delta^{(0)}$, form $r^{(0)} = -F - J\delta^{(0)}$, where $F = F(u_n)$ and $J = J(u_n)$.
- Compute $\beta = \|r^{(0)}\|_2$ and $v_1 = r^{(0)}/\beta$.
- For $j = 1, 2, \cdots$, do:
  (a) Form Jv_j and orthogonalize it against the previous $v_1, \cdots, v_j$ via

$$\begin{aligned} h_{i,j} &= (Jv_j, v_i), \quad i = 1, 2, \cdots, j, \\ \hat{v}_{j+1} &= Jv_j - \sum_{i=1}^{j} h_{i,j} v_i \\ h_{j+1,j} &= \|\hat{v}_{j+1}\|_2, \quad \text{and} \\ v_{j+1} &= \hat{v}_{j+1}/h_{j+1,j}. \end{aligned} \tag{2.7}$$

  (b) Compute the residual norm $\rho_j = \|F + J\delta^{(j)}\|_2$, of the solution $\delta^{(j)}$ that would be obtained if we stopped at this step.
  (c) If $\rho_j \leq \epsilon_n$ set $m = j$ and go to (3).

(3) *Form the approximate solution:*
Arnoldi: Define H_m to be the $m \times m$ (Hessenberg) matrix whose nonzero entries are the coefficients h_{ij}, $1 \leq i \leq j$, $1 \leq j \leq m$ and define $V_m \equiv [v_1, v_2, \cdots, v_m]$.
- Find the vector y_m that solves the linear system $H_m y = \beta e_1$, where $e_1 = [1, 0, \cdots, 0]^T$.
- Compute $\delta^{(m)} = \delta^{(0)} + z^{(m)}$, where $z^{(m)} = V_m y_m$, and $u_{n+1} = u_n + \delta^{(m)}$.

GMRES: Define $\bar{H}_m$ to be the $(m+1) \times m$ (Hessenberg) matrix whose nonzero entries are the coefficients h_{ij}, $1 \leq i \leq j+1$, $1 \leq j \leq m$ and define $V_m \equiv [v_1, v_2, \cdots, v_m]$.
- Find the vector y_m that minimizes $\|\beta e_1 - \bar{H}_m y\|_2$ over all vectors y in $\mathbf{R}^m$, where $e_1 = [1, 0, \cdots, 0]^T$.
- Compute $\delta^{(m)} = \delta^{(0)} + z^{(m)}$ where $z^{(m)} = V_m y_m$, and $u_{n+1} = u_n + \delta^{(m)}$.

(4) *Stopping test:* If u_{n+1} is determined to be a good enough approximation to a root of (2.1), then stop, else set $u_n \leftarrow u_{n+1}$, $n \leftarrow n+1$, choose a new tolerance ϵ_n, and go to (2).

Therefore, in both Arnoldi and GMRES the outer iteration is of the form $u_{n+1} = u_n + \delta^{(m)}$ where $\delta^{(m)} = \delta^{(0)} + z^{(m)}$, with

$$z^{(m)} = V_m y_m,$$

and y_m is either the solution of an $m \times m$ linear system, for Arnoldi, or the solution of an $(m+1) \times m$ least squares problem for GMRES.

Steps (2) and (3) of the above algorithm are precisely the Arnoldi (GMRES) method for solving the linear system $J\delta = -F$ (see [22] and [24]). Each outer loop of the above algorithm, consisting of steps (2), (3), and (4), is divided into two main stages. The first stage is an Arnoldi process, which builds an orthonormal basis $V_m = [v_1, v_2, \cdots, v_m]$ of the Krylov subspace K_m. If we denote by V_j the $N \times j$ matrix with column vectors $v_1, v_2, \cdots, v_j$, then it follows immediately from (2.7) that (see also [22] and [24])

$$JV_m = V_m H_m + \hat{v}_{m+1} e_m^T, \tag{2.8}$$

where $e_m = [0, \cdots, 0, 1]^T \in \mathbf{R}^m$. This relation, which can be rewritten as

$$JV_m = V_{m+1} \bar{H}_m, \tag{2.9}$$

is crucial in the development of the Arnoldi and GMRES methods.

Step (3) of Newton–GMRES computes the approximate solution $\delta^{(m)}$ in $\delta^{(0)} + K_m$ that solves (2.6). This is accomplished by first letting $z = V_m y$ for $y \in \mathbf{R}^m$. Then $\|r^{(0)} - Jz\|_2 = \|\beta v_1 - JV_m y\|_2$. Using (2.9), we have

$$\begin{aligned} \|\beta v_1 - JV_m y\|_2 &= \|V_{m+1}(\beta e_1 - \bar{H}_m y)\|_2 \\ &= \|\beta e_1 - \bar{H}_m y\|_2, \end{aligned}$$

since V_{m+1} has orthonormal columns. We denote by y_{GM} the solution of the minimization problem

$$\min_{y \in \mathbf{R}^m} \|\beta e_1 - \bar{H}_m y\|_2. \tag{2.10}$$

Then the optimal δ is given by $\delta^{(m)} = \delta^{(0)} + V_m y_{GM}$. Note that (2.10) is a least squares problem of size $m + 1$, and its coefficient matrix is upper Hessenberg. The next iterate u_{n+1} is then computed at the end of step (3), by adding $\delta^{(m)}$. Finally, the stopping criteria in step (4) will be discussed in the numerical testing section.

For simplicity, we have omitted several details on the practical implementation of the above methods, which are discussed at length in [22], [4], and [24]. For example, the residual norm ρ_j referred to in step (2) of the algorithms does not require the computation of the approximate solution $\delta^{(j)}$ at every step. Instead an inexpensive formula, which evaluates ρ_j, is updated at each step while the factorization of the Hessenberg matrix H_m or $\bar{H}_m$ is updated (see [4] and [24] for details).

The Arnoldi algorithm is theoretically equivalent to the Conjugate Gradient method when J is symmetric and positive definite, and to the Lanczos method for solving linear systems when J is symmetric [22]. The GMRES algorithm is theoretically equivalent to GCR [8] and to ORTHODIR [15] but is less costly both in terms of storage and arithmetic [24]. For a synthesis and general description of available conjugate gradient type methods see [23]. A comparison of the cost of each step of these algorithms shows that for large enough m, GMRES costs about 1/3 less than GCR/ORTHOMIN in arithmetic, whereas storage is roughly divided by a factor of two. Another appealing property of GMRES is that in exact arithmetic, the method does not break down or, to be more accurate, it can only break down when it delivers the exact solution [24].

We note that if either algorithm solves the linear system $J\delta = -F(u)$ exactly, or rather with sufficient accuracy by taking (for example) m sufficiently large, then it is clear that the resulting algorithm is nothing but Newton's method, in which the Jacobian linear systems are solved by either Arnoldi or GMRES.

Perhaps one of the most important aspects of the above Krylov methods is that the Jacobian matrix J is never needed explicitly. The only operations with the Jacobian matrix J that are required from the Arnoldi process are matrix-vector multiplications $w = Jv$, which can be approximated by

$$J(u)v \approx \frac{F(u + \sigma v) - F(u)}{\sigma}, \tag{2.11}$$

where u is the point at which the Jacobian is being evaluated and σ is some carefully chosen small scalar. The idea of exploiting the above approximation is not new and was extensively used in the context of ODE methods [3]–[5], [11], [17], in eigenvalue calculations [9], [16] and is quite common in nonlinear equation solution methods and optimization methods (see, for example, [12],[19],[27]).

Another aspect of the above algorithms we have not yet considered is the ability to use restarting in the (linear) Krylov methods. In a typical implementation of the above Krylov methods, a maximum value of m is dictated by storage considerations. If we let $m_{\max}$ be this value, then it is possible that $m = m_{\max}$ in the Arnoldi process, and yet ρ_m is still greater than ϵ_n. In this case, we can set $\delta^{(0)}$ equal to $\delta^{(m)}$ and restart the Arnoldi process, effectively restarting the Krylov method. The convergence of such a procedure is not always guaranteed, but the idea seems to work well in practice. We note that for lack of a better initial guess we use $\delta^{(0)} = 0$ on the first (and possibly only) pass through the Arnoldi process at each stage of the Newton iteration. It is only when restarting that $\delta^{(0)}$ will be nonzero. As will be seen below, it will also be important to choose the tolerance ϵ_n at each step of the Newton iteration.

Finally, we note that as m becomes large, a considerable amount of the work involved is in making the vector v_{j+1} orthogonal to all the previous vectors $v_1, \cdots, v_j$. Saad [22] and Brown and Hindmarsh [4] have proposed incomplete versions of Arnoldi and GMRES, respectively, in which the vector v_{j+1} is only required to be orthogonal to the previous p vectors , $v_{j-p+1}, \cdots, v_j$. Equations (2.8) and (2.9) still hold in this case but the basis $V_{m+1} = [v_1, \cdots, v_{m+1}]$ is only partially orthogonal in the sense defined above. These algorithms are referred to as the *Incomplete Orthogonalization Method* (IOM) and IGMRES, respectively, and can be more cost effective than the complete methods on some problems. See [22] and [4] for details.

3. Linesearch backtracking techniques. Newton's method by itself may often fail to converge if the initial guess u_0 is far away from a root of (1.1). To enhance the robustness of the nonlinear Krylov algorithms considered in the previous section we will consider two modifications of these methods. In this section, we will consider a global strategy based on a linesearch backtracking procedure, and then in the next section we will investigate a model trust region approach.

Dennis and Schnabel [7] suggest using a global strategy for finding a root u_* of (1.1) that is based upon a globally convergent method for the problem

$$\min_{u \in \mathbf{R}^N} f(u) = \frac{1}{2} F(u)^T F(u). \tag{3.1}$$

A *descent direction* for f at the current approximation u is any vector p such that

$$\nabla f(u)^T p < 0,$$

where $\nabla f(u) = (\partial f/\partial u_1(u), \cdots, \partial f/\partial u_N(u))^T$. An easy calculation shows that

$$\nabla f(u) = J(u)^T F(u),$$

and so p is a descent direction for f at u if

$$F(u)^T J(u) p < 0.$$

For such a direction, one can show that there exists a certain $\lambda_0 > 0$ such that $f(u + \lambda p) < f(u)$ for all $0 < \lambda \leq \lambda_0$.

If $\bar{\delta}$ is an approximate solution of (2.2), with $F = F(u)$ and $J = J(u)$, then

$$F^T J \bar{\delta} = -F^T F - F^T \bar{r}, \tag{3.2}$$

where $\bar{r} = -F - J\bar{\delta}$ is the residual associated with $\bar{\delta}$. Thus, $\bar{\delta}$ will be a descent direction for f at u whenever $|F^T \bar{r}| < F^T F$. In particular, if $\|\bar{r}\|_2 < \|F\|_2$, then $\bar{\delta}$ is

a descent direction. We have the following results regarding the existence of descent directions when using the Arnoldi and GMRES methods.

PROPOSITION 3.1. *Let u be the current Newton–Arnoldi iterate, $F \equiv F(u)$ and $J \equiv J(u)$. Assume J is nonsingular. Let $\delta^{(m)} = -V_m H_m^{-1} V_m^T F$ be the direction provided by the Arnoldi method assuming the initial guess $\delta^{(0)} = 0$. If $\delta^{(m)}$ exists, then it is a descent direction for f at u, and*

$$F^T J \delta^{(m)} = -F^T F, \tag{3.3}$$

for any $m = 1, \cdots, N$.

Proof. (See Brown [2, Thm. 3.5]). This result follows from the equality (3.2) and the fact that the residual vector in Arnoldi's method is orthogonal to the Krylov subspace, and in particular to its first basis vector, which is F up to a constant factor. □

PROPOSITION 3.2. *Let u be the current Newton–*GMRES *iterate, $F = F(u)$ and $J = J(u)$. Assume J is nonsingular. Let $\delta^{(m)} = V_m y_{GM}$ be the direction provided by the* GMRES *method assuming the initial guess $\delta^{(0)} = 0$. If $\delta^{(m)} \neq 0$, then $\delta^{(m)}$ is a descent direction for f at u, and*

$$F^T J \delta^{(m)} = -F^T F + \rho_m^2, \tag{3.4}$$

for any $m = 1, \cdots, N$, where ρ_m is the residual norm achieved at the mth step of the GMRES *algorithm.*

Proof. We first prove (3.4). According to (3.2) we only have to show that $\rho_m^2 = -F^T r^{(m)}$ with $r^{(m)} = -F - J\delta^{(m)}$. By definition (see (2.6)), *GMRES* minimizes $\|F + Jz\|_2$ for z in the Krylov subspace K_m. It is known in this case that the residual vector corresponding to the minimizer is orthogonal to JK_m (see, e.g., [23]). Therefore, we have

$$(r^{(m)}, F) = (r^{(m)}, -r^{(m)} - J\delta^{(m)}) = -\rho_m^2,$$

which establishes (3.4). The fact that $\delta^{(m)}$ is a descent direction in this case, has been shown in Theorem 3.7 of [2]. It can also be shown from (3.4) and the definition (2.6), from which we get that $\rho_m < \rho_0 = \|F\|_2$, since $\delta^{(m)} \neq 0$. □

We also have the following result regarding the existence of descent directions in the subspace K_m for f at u.

PROPOSITION 3.3. *Let u be the current Newton–*GMRES *iterate, $F = F(u)$ and $J = J(u)$. Assume J is nonsingular. Let $\delta^{(m)} = V_m y_{GM}$ be the direction provided by the* GMRES *method assuming the initial guess $\delta^{(0)} = 0$. Then there exist descent directions in the subspace K_m for the function f at u if and only if*

$$\rho_m = \|F + J\delta^{(m)}\|_2 < \|F\|_2 \tag{3.5}$$

holds.

Proof. This result is clear from Proposition 3.2 and from the fact that the GMRES iterate solves the minimization problem (2.6). □

The assumption that J is nonsingular in the above propositions is necessary, since Arnoldi and GMRES are only guaranteed to converge for nonsingular systems. When J is singular, the Arnoldi process can break down before providing any useful information, since Jv_i might vanish for some i. Furthermore, while Arnoldi's method will converge in at most N iterations for any nonsingular J, the Hessenberg matrix

H_m may be singular for some $m < N$, and the mth Arnoldi iterate may not exist as a result, i.e., there may be no solution to the system $H_m y = \beta e_1$. See [4], [22], and [24] for more details.

Condition (3.5) means that the residual norm in GMRES must be reduced strictly. It holds whenever the Jacobian matrix is positive real and at least one step of GMRES is performed, i.e., $m \geq 1$ (see Brown [2] and Elman [8]). The condition that J be positive real at every step is too strong a condition to require. A milder condition is to assume that the dimension m in Arnoldi or GMRES is large enough to ensure that the final residual is reduced by a factor of at least η, where η is a scalar < 1. In other words,

$$\|J\delta + F\|_2 \leq \eta \|F\|_2, \tag{3.6}$$

where $\eta < 1$, and δ is the Arnoldi or GMRES iterate.

The practicality of the assumption that m can be chosen as large as necessary to guarantee that (3.6) holds is questionable. It is known that as m gets large, (3.6) will eventually be fulfilled. The problem is that with m too large, computational cost and storage become too high. An alternative would be to perform restarting within the (linear) Arnoldi or GMRES algorithm itself. In this case, however, the above results (3.3) and (3.4) do not hold, since their proofs rely on the assumption that $\delta^{(0)} = 0$, while restarting has the same effect as making the initial guess nonzero in Arnoldi or GMRES. To derive similar results, we go back to (3.2), which becomes

$$\nabla f(u)^T \delta^{(m)} = -F^T F - F^T r^{(m)}. \tag{3.7}$$

Thus, for both GMRES and Arnoldi, it suffices that $\delta^{(m)}$ will be a descent direction whenever $\|r^{(m)}\|_2 < \|F\|_2$, a condition that has already been seen at the beginning of this section.

We also need to obtain alternative expressions for $\nabla f(u)^T \delta^{(m)}$ for computational purposes. It is easy to show (see, e.g., [22]) that for Arnoldi's method the residual vector satisfies

$$r^{(m)} = -F - J\delta^{(m)} = -(e_m^T y_m)\hat{v}_{m+1},$$

which yields from (3.7),

$$\nabla f(u)^T \delta^{(m)} = -F^T F + (e_m^T y_m) F^T \hat{v}_{m+1}. \tag{3.8}$$

Note that when $\delta^{(0)} = 0$ the last term in the above expression vanishes, since then $F = -\beta v_1$, and the columns of V_m are orthogonal to $\hat{v}_{m+1}$. For GMRES we may use (3.7) but we need to compute $r^{(m)}$. We have,

$$\begin{aligned} r^{(m)} &= -F - J\delta^{(m)} \\ &= r^{(0)} - JV_m y_m \\ &= V_{m+1}(\beta e_1 - \bar{H}_m y_{GM}), \end{aligned} \tag{3.9}$$

which does not involve function evaluations. It can also be shown that (see [4] for details)

$$\beta e_1 - \bar{H}_m y_{GM} = \beta q_{m+1} q_{m+1}^T e_1,$$

where q_{m+1} is the last column of the Q matrix in the QR factorization of $\bar{H}_m$.

Our backtracking procedure will be based upon the ideas presented by Dennis and Schnabel [7]. Given the current Newton iterate $u = u_n$ and a descent direction p, we want to take a step in the direction of p that yields an acceptable u_{n+1}. We will define a step $\delta = \lambda p$ to be *acceptable* if the following Goldstein–Armijo [7] conditions are met:

$$f(u + \lambda p) \leq f(u) + \alpha\lambda \nabla f(u)^T p, \text{ and} \tag{3.10}$$

$$f(u + \lambda p) \geq f(u) + \beta\lambda \nabla f(u)^T p, \tag{3.11}$$

for given scalars α, β satisfying $0 < \alpha < \beta < 1$. These two conditions are commonly referred to as the α- and β-conditions, respectively. For a given descent direction p, the next result shows that there exist points $u + \lambda p$ satisfying (3.10) and (3.11).

THEOREM 3.4. *Let $F : \mathbf{R}^N \to \mathbf{R}^N$ be continuously differentiable on $\mathbf{R}^N$. Let $f(u) = \frac{1}{2}F(u)^T F(u)$, and u, p in $\mathbf{R}^N$ such that $\nabla f(u)^T p < 0$. Then given $0 < \alpha < \beta < 1$, there exist $\lambda_u > \lambda_\ell > 0$ such that $u + \lambda p$ satisfies* (3.10) *and* (3.11) *for any $\lambda \in (\lambda_\ell, \lambda_u)$.*

Proof. This is essentially Theorem 6.3.2 of Dennis and Schnabel [7, p. 120]. □

We next present the particular backtracking algorithm we have chosen to use. The selection procedure for λ is modeled after that in [7].

Algorithm: Linesearch Backtrack

(1) Choose $\alpha \in (0, \frac{1}{2})$ and $\beta \in (\frac{1}{2}, 1)$.

(2) Given u_n the current Newton iterate, calculate $p = \delta^{(m)}$, where $\delta^{(m)} = \delta^{(0)} + z^{(m)}$, and $z^{(m)} = V_m y_m$. Here, y_m is calculated using either the Arnoldi or GMRES method (with or without restarting), and it is assumed that (3.6) holds with $\delta = \delta^{(m)}$.

(3) Calculate $\nabla f(u_n)^T p$ using the appropriate choice(s) from equations (3.3), (3.4), (3.7), (3.8), and (3.9).

(4) Find an acceptable new iterate $u_{n+1} = u_n + \lambda p$. First, set $\lambda = 1$. Define $u(\lambda) = u_n + \lambda p$.

  (a) If $u(\lambda)$ satisfies (3.10) and (3.11), then exit. If not, then continue.

  (b) If $u(\lambda)$ satisfies (3.10), but not (3.11), and $\lambda \geq 1$, set $\lambda \leftarrow 2\lambda$ and go to (a).

  (c) If $u(\lambda)$ satisfies (3.10) only and $\lambda < 1$, or $u(\lambda)$ does not satisfy (3.10) and $\lambda > 1$, then

    (c.1) If $\lambda < 1$, define $\lambda_{lo} = \lambda$ and λ_{hi} = last previously attempted value of λ. If $\lambda > 1$, define λ_{lo} = last previously attempted value of λ and $\lambda_{hi} = \lambda$. In both cases, $u(\lambda_{lo})$ satisfies (3.10) but not (3.11), $u(\lambda_{hi})$ does not satisfy (3.10), and $\lambda_{lo} < \lambda_{hi}$.

    (c.2) Find $\lambda \in (\lambda_{lo}, \lambda_{hi})$ such that $u(\lambda)$ satisfies (3.10) and (3.11) using successive linear interpolation.

  (d) Otherwise ($u(\lambda)$ does not satisfy (3.10) and $\lambda \leq 1$), decrease λ by a factor between 0.1 and 0.5 as follows:

    (d.1) Select the new λ such that $u(\lambda)$ is the minimizer of the one-dimensional quadratic interpolant passing through $f(u_n)$, $f'(u_n) = \nabla f(u_n)^T p$ and $f(u_n + \lambda p)$. Then take the maximum of this new λ and 0.1 as the actual value used. (One can show theoretically that the new λ value so chosen will be less than or equal to one-half the previous value.)

    (d.2) Go to step (b).

We note that the current algorithm will likely break down if the Jacobian J is singular. For now, all that can be done in this case is to restart the iteration with a different initial guess u_0.

It remains to discuss the choice of ϵ_n. From the results in [6], choosing $\epsilon_n = \eta\|F(u_n)\|_2$, where $0 < \eta < 1$, guarantees the linear convergence of the Newton–Krylov iteration for a good enough initial guess. If η is replaced by a sequence η_n decreasing to zero and satisfying $0 \leq \eta_n < 1$ for all n, then the iteration converges superlinearly. It is possible to show that for this second choice of ϵ_n the full Arnoldi or GMRES step will be admissible near a root (i.e., that conditions (3.10) and (3.11) will be satisfied with $\lambda = 1$). However, a finer analysis may indicate that the looser tolerance may in fact give convergence of the iteration.

4. Model trust region techniques. In this section we will propose a model trust region strategy in connection with the Newton–GMRES algorithm. In particular, we will consider a dogleg strategy based on Powell's hybrid method [21].

Let u be the current approximate solution of (1.1). The effect of using a Krylov method to solve the Newton equations (1.2) approximately is to take a step from u of the form $u + \delta$, where δ is in the affine subspace $\delta^{(0)} + K_m$. If $V_m = [v_1, \cdots, v_m]$ is an orthonormal basis for K_m, and the initial guess $\delta^{(0)} = 0$, then $\delta = V_m y$, for some $y \in \mathbf{R}^m$, and we have a step of the form $u + V_m y$. Hence, we are effectively restricting our search directions from u to be in the subspace K_m.

4.1. Global strategy with restriction to a subspace. Our global strategy will again be based upon finding a local minimum of the real-valued function $f(u) = \frac{1}{2}F(u)^T F(u)$. Thus, we want to solve the minimization problem

$$\min_{y \in \mathbf{R}^m} f(u + V_m y).$$

Letting $g(y) = f(u + V_m y)$, we then have

$$\nabla g(y) = (J(u + V_m y)V_m)^T F(u + V_m y),$$

and in particular that

$$\nabla g(0) = (JV_m)^T F,$$

where $F = F(u)$ and $J = J(u)$.

If we use $F + JV_m y$ as a linear model of $F(u + V_m y)$, then a natural quadratic model for g is

$$\hat{g}(y) = \frac{1}{2}\|F + JV_m y\|_2^2. \tag{4.1}$$

Letting $B_m = V_m^T J^T J V_m$, we have

$$\hat{g}(y) = \frac{1}{2}F^T F + F^T J V_m y + \frac{1}{2}y^T B_m y,$$

where B_m is symmetric and positive semidefinite, and $\nabla\hat{g}(0) = \nabla g(0)$. If J is nonsingular, then B_m is positive definite, since V_m has orthonormal columns. Our *model trust region* approach will be based upon trying to find a solution of the problem

$$\min_{\|y\|_2 \leq \tau} \hat{g}(y), \quad y \in \mathbf{R}^m, \tag{4.2}$$

where τ is an estimate of the maximum length of a successful step we are likely to be able to take from u. It is also a measure of the size of the region in which the local quadratic model $\hat{g}(y)$ closely agrees with the function $g(y)$. The solution to (4.2) is given in the next lemma.

LEMMA 4.1. *Let $\hat{g}(y)$ be defined by* (4.1), *and assume that J is nonsingular. Then problem* (4.2) *is solved by*

$$y_m(\mu) = (B_m + \mu I)^{-1} d_m, \tag{4.3}$$

where $d_m = -\nabla \hat{g}(0)$, for the unique μ such that $\|y_m(\mu)\|_2 = \tau$, unless $\|y_m(0)\|_2 \leq \tau$, in which case $y_m(0) = B_m^{-1} d_m$ is the solution. Furthermore, for any $\mu \geq 0$, $\delta(\mu) = V_m y_m(\mu)$ defines a descent direction for $f(u) = \frac{1}{2} F(u)^T F(u)$ from u, as long as $d_m \neq 0$.

Proof. Since J being nonsingular implies that B_m is positive definite, (4.3) follows from Lemma 6.4.1 of Dennis and Schnabel [7, p. 131] and $y_m(\mu)$ is the unique solution to (4.2). It therefore only remains to show that $\delta(\mu) = V_m y_m(\mu)$ is a descent direction for $f(u)$ at u, for all $\mu \geq 0$. Recall that p is a descent direction for f at u if

$$\nabla f(u)^T p < 0,$$

or since $\nabla f(u) = J^T F$, if

$$F^T J p < 0.$$

For $p = \delta(\mu)$, we have

$$\begin{aligned} F^T J \delta(\mu) &= F^T J V_m (B_m + \mu I)^{-1} d_m \\ &= ((J V_m)^T F)^T (B_m + \mu I)^{-1} d_m \\ &= -d_m^T (B_m + \mu I)^{-1} d_m \\ &< 0, \end{aligned}$$

since $d_m = -\nabla \hat{g}(0) = \nabla g(0) \neq 0$, and since $B_m + \mu I$ is positive definite for all $\mu \geq 0$. □

Since there is no finite method of determining μ such that $\|y_m(\mu)\|_2 = \tau$ when $\tau < \|B_m^{-1} d_m\|_2$, we only approximately solve (4.2). The *dogleg strategy* of Powell [21] makes a piecewise linear approximation to the curve $y_m(\mu)$, and takes $\hat{y}_m$ as the point on this curve for which $\|\hat{y}_m\|_2 = \tau$. We then define $u_{n+1} = u_n + \hat{\delta}$, where $\hat{\delta} = V_m \hat{y}_m$. If the iterate u_{n+1} is acceptable, in the sense that some α-condition (3.10) is satisfied, we proceed to the next step, while if not, a new value of the trust region size τ is chosen, and the procedure is repeated.

4.2. Global strategies for GMRES. The preceding discussion is independent of the choice of the subspace K_m, i.e., the results are valid for any subspace K_m of dimension m with orthonormal basis given by the columns of V_m. In fact, the basis vectors v_i $(i = 1, \cdots, m)$ need not even form an orthogonal basis for K_m. We next turn our attention to the case in which K_m and V_m are generated using the GMRES algorithm. Here, the relation (2.9) allows an easy reformulation of the minimization problem (4.2). First, note that when the initial guess is $\delta^{(0)} = 0$ and m steps of the Arnoldi process have been taken in the Newton–GMRES algorithm we have

$$d_m = -\nabla \hat{g}(0) = -(J V_m)^T F = -\beta \bar{H}_m^T e_1. \tag{4.4}$$

This direction is referred to as the *steepest descent direction* for $\hat{g}(y)$ at $y = 0$, and is the same as that for $g(y)$. Next,

$$\begin{aligned} B_m &= (JV_m)^T JV_m \\ &= (V_{m+1}\bar{H}_m)^T V_{m+1}\bar{H}_m \\ &= \bar{H}_m^T \bar{H}_m. \end{aligned} \tag{4.5}$$

Here we have used the relation (2.9), which is satisfied by the Arnoldi vectors v_i. Thus, $\hat{g}(y)$ can be rewritten as

$$\hat{g}(y) = \frac{1}{2}F^T F + \beta e_1^T \bar{H}_m y + \frac{1}{2} y^T \bar{H}_m^T \bar{H}_m y. \tag{4.6}$$

Furthermore, the fact that J is of full rank implies that $\bar{H}_m$ is also of full rank by (2.9) and the orthogonality of V_m and V_{m+1}. Note that this is true even when $h_{m+1,m} = 0$, because in this case the relation (2.9) reduces to $JV_m = V_m H_m$, with H_m nonsingular and $\bar{H}_m^T \bar{H}_m = H_m^T H_m$.

Minimizing $\hat{g}(y)$ in the steepest descent direction amounts to minimizing

$$\hat{g}(\alpha d_m) = \frac{1}{2}F^T F + \alpha\beta e_1^T \bar{H}_m d_m + \frac{\alpha^2}{2}\|\bar{H}_m d_m\|_2^2.$$

The optimal value for α is

$$\alpha_{\text{opt}} = \frac{\|d_m\|_2^2}{\|\bar{H}_m d_m\|_2^2},$$

which is defined as long as $d_m \neq 0$, since $\bar{H}_m$ has full column rank. We refer to the point where $\hat{g}(\alpha d_m)$ assumes its minimum value as the *Cauchy Point*, and write

$$y_{CP} = \alpha_{\text{opt}} d_m = -\beta \frac{\|d_m\|_2^2}{\|\bar{H}_m d_m\|_2^2} \bar{H}_m^T e_1. \tag{4.7}$$

We note that GMRES gives the global minimizer of $\hat{g}(y)$, and we write

$$y_{GM} = -(\bar{H}_m^T \bar{H}_m)^{-1} \beta \bar{H}_m^T e_1 = (\bar{H}_m^T \bar{H}_m)^{-1} d_m. \tag{4.8}$$

Note that in practice the above formula will never be used explicitly. Numerically, it is more satisfactory to compute y_{GM} as the solution of the the least squares problem $\min \|\bar{H}_m y + \beta e_1\|_2$ over the variable y. We should point out here that when $h_{m+1,m} = 0$ the determination of y_{GM} in this manner does not cause any difficulty. In fact, in this situation we will have what is called a *happy breakdown* in GMRES, in that $\delta = V_m y_{GM}$ becomes the exact solution to the linear system $J\delta = -F$, i.e., we obtain a full Newton step instead of an inexact Newton step. The GMRES solution and the Arnoldi solution are then identical.

In order to use the dogleg strategy, we have seen that the vector d_m must be nonzero. Since d_m is the steepest descent direction within the Krylov subspace, a null value would indicate that there are no descent directions from f at u within the subspace K_m. Note that (4.8) implies $y_{GM} = 0 \Leftrightarrow d_m = 0$. Hence, if we require m to be large enough so that condition (3.6) holds, then necessarily y_{GM} will be nonzero, and the dogleg strategy can be used. This will be assumed in the dogleg algorithm presented below.

We next describe the dogleg algorithm for $\hat{g}(y)$. Suppose we are in a situation where the full GMRES step is unsatisfactory. This indicates that the quadratic model $\hat{g}(y)$ does not adequately model $g(y)$ in a region containing the full GMRES step. The linesearch algorithm of the previous section would take a step in the same direction, but of shorter length. In the dogleg strategy, we first choose a shorter steplength, and then use the full m-dimensional quadratic model $\hat{g}(y)$ to choose the new direction. The curve $y_m(\mu)$ given by (4.3) is approximated by the piecewise linear curve from zero to y_{CP}, and from y_{CP} to y_{GM}. As indicated above, given a trust region size τ, we then find the point $\hat{y}_m$ on the dogleg curve with $\|\hat{y}_m\|_2 = \tau$.

The only remaining parts of the algorithm to discuss are the decision process for an acceptable new iterate u_{n+1} and the selection of the trust region size τ. We will again base our strategy on the ideas presented in [7]. The condition for accepting u_{n+1} is (3.10), namely,

$$f(u_n + \hat{\delta}) \leq f(u_n) + \alpha \nabla f(u_n)^T \hat{\delta},$$

where $\hat{\delta} = V_m \hat{y}_m$. In this case, the relations (3.3) and (3.4) do not apply, since $\hat{\delta}$ is not necessarily the Arnoldi or GMRES direction. However, for any $\delta = V_m y$, we have

$$\nabla f(u_n)^T \delta = F^T J \delta = F^T J V_m y = F^T V_{m+1} \bar{H}_m y = -\beta e_1^T \bar{H}_m y. \tag{4.9}$$

Hence, it will be easy to determine if (3.10) is satisfied for any u_{n+1} of the form $u_n + \hat{\delta}$ without even computing the direction $\hat{\delta}$.

The trust region size will be adjusted based upon a comparison of the two values

$$\Delta f \equiv f(u_{n+1}) - f(u_n) = g(\hat{y}_m) - g(0),$$

which is the *actual reduction* in the function f, and

$$\Delta f_{\text{pred}} \equiv \hat{g}(\hat{y}_m) - g(0),$$

which is the *predicted reduction* in the function f. Our dogleg algorithm is then as follows.

Algorithm: Dogleg

(1) Choose $\alpha \in (0, \frac{1}{2})$.

(2) Given u_n, the current Newton iterate, calculate $\delta^{GM} = V_m y_{GM}$. Here, y_{GM} is calculated using the GMRES method (without restarting) with initial guess $\delta^{(0)} = 0$, and it is assumed m is large enough so that (3.6) holds with $\delta = \delta^{GM}$.

(3) Given τ, the current trust region size, calculate $\hat{y}_m$, the point on the dogleg curve for which $\|\hat{y}_m\|_2 = \tau$. Then calculate $u_{n+1} = u_n + V_m \hat{y}_m$. If u_{n+1} is acceptable, then go to step (5).

(4) If u_{n+1} is not acceptable, then do one of the following:

   (a) If τ has been doubled during this iteration, then set u_{n+1} equal to its last accepted value and set $\tau \leftarrow \tau/2$. Then continue to the next Newton iteration. If not:

   (b) Determine a new τ by using the minimizer of the one-dimensional quadratic interpolating $f(u_n)$, $f(u_{n+1})$, and the directional derivative of f at u_n in the direction $\hat{\delta} = V_m \hat{y}_m$. Letting λ be the value for which $u_n + \lambda \hat{\delta}$ is this minimizer, set $\tau \leftarrow \lambda \|\hat{\delta}\|_2$, but constraining it to be between 0.1 and 0.5 of the old τ. Then go to step (3).

(5) For an acceptable u_{n+1}, calculate Δf and Δf_{pred}. Then do one of the following:
   (a) If Δf and Δf_{pred} agree to within relative error 0.1, and τ has not been decreased during this iteration, set $\tau \leftarrow 2 * \tau$, and go to step (3). If not:
   (b) If $\Delta f > 0.1 * \Delta f_{\text{pred}}$ set $\tau = \tau/2$, or if $\Delta f < 0.75 * \Delta f_{\text{pred}}$, set $\tau = 2 * \tau$. Otherwise, do not change τ. Then continue to the next Newton iteration. (Note that here both Δf and Δf_{pred} are negative.)

We note that this algorithm will also likely break down if the Jacobian J is singular. As with the linesearch technique, all that can be done for now is to restart the iteration with a different initial guess u_0. We also set $\epsilon_n = \eta_n \|F(u_n)\|_2$ as before.

Although we have based our trust region algorithm on the dogleg strategy outlined in [7], we could alternatively have based our approach on one of the related algorithms for unconstrained optimization presented by Moré [18] and Gay [10]. In general, for unconstrained optimization problems the matrix B_m can have negative eigenvalues. In this case, Gay [10] has shown that the solution of (4.2) is still a y_* value satisfying $(B_m + \mu I)y_* = d_m$ for some $\mu > 0$ such that $B_m + \mu I$ is positive semidefinite.

4.3. Restarting procedures. One can allow restarting in the (linear) Krylov method in the above algorithm. Equivalently, we would like to show how to use a nonzero initial vector $\delta^{(0)}$ in the algorithm. In this case, the step δ from u would normally have the form $\delta = \delta^{(0)} + V_m y$. However, with this choice $g(y) = f(u + \delta) = f(u + \delta^{(0)} + V_m y)$ would not necessarily be close to $f(u)$ for small y, as it should be. That is, $g(0) \neq f(u)$ in general. For this reason, the contribution to the step from the initial guess must also be variable. This has the effect of enlarging the dimension of the minimization problem (4.2). For $\delta^{(0)}$ a nonzero initial guess, let $\bar{y} = (y, t)^T \in \mathbf{R}^{m+1}$ and try taking a step from u of the form $u+\delta$, where $\delta = [V_m, \delta^{(0)}]\bar{y}$. (Here, we assume $V_m = [v_1, \cdots, v_m]$ has been generated by taking m steps of the Arnoldi process.) Then we have

$$F(u + \delta) \approx F + J[V_m, \delta^{(0)}]\bar{y}.$$

Letting $W = [V_m, \delta^{(0)}]$, the local quadratic model for $g(\bar{y}) = f(u + W\bar{y}) = \frac{1}{2}||F(u + W\bar{y})||_2^2$ is now given by

$$\hat{g}(\bar{y}) = \frac{1}{2}\|F + JW\bar{y}\|_2^2.$$

Since $r^{(0)} = -F - J\delta^{(0)}$, we can write

$$F + JW\bar{y} = (1 - t)F - t\, r^{(0)} + V_{m+1}\bar{H}_m y,$$

using the fact that $JV_m = V_{m+1}\bar{H}_m$. This then makes it possible to evaluate $\hat{g}(\bar{y})$ without the need for the Jacobian matrix J.

For this modified quadratic model, the *steepest descent direction* is given by

$$\begin{aligned} d &= -\nabla\hat{g}(0) = -(JW)^T F \\ &= -[JV_m, J\delta^{(0)}]^T F \\ &= -[V_{m+1}\bar{H}_m, J\delta^{(0)}]^T F \\ &= \begin{pmatrix} -H_m^T V_{m+1}^T F \\ (F + r^{(0)})^T F \end{pmatrix} \in \mathbf{R}^{m+1}. \end{aligned}$$

Next, the *approximate Newton step* is found using a relationship between the Newton step and the steepest descent direction similar to the situation earlier without restarting, namely,

$$s = B^{-1}d,$$

where $B = \nabla^2\hat{g}(0)$ is the Hessian of $\hat{g}$. An easy calculation gives

$$B = (JW)^T(JW) = \begin{pmatrix} \bar{H}_m^T\bar{H}_m & v \\ v^T & a \end{pmatrix},$$

letting $v = -\bar{H}_m^T V_{m+1}^T(F + r^{(0)})$ and $a = \|F + r^{(0)}\|_2^2$. To calculate s, first note that we already have $\bar{H}_m = QR$, the QR-factorization of $\bar{H}_m$. Hence,

$$B = \begin{pmatrix} R^TR & v \\ v^T & a \end{pmatrix}.$$

Letting

$$R = \begin{pmatrix} \bar{R} \\ 0 \cdots 0 \end{pmatrix},$$

where $\bar{R} \in \mathbf{R}^{m\times m}$, we have $R^TR = \bar{R}^T\bar{R}$, which is essentially a Cholesky decomposition of $\bar{H}_m^T\bar{H}_m$. Note that the matrix $\bar{R}$ is always nonsingular for J nonsingular, regardless of the value of $h_{m+1,m}$ in the Arnoldi process. The Cholesky decomposition of B is then easily obtained as

$$B = \begin{pmatrix} \bar{R}^T & 0 \\ w^T & b \end{pmatrix}\begin{pmatrix} \bar{R} & w \\ 0 & b \end{pmatrix} \equiv C^TC, \tag{4.10}$$

where $w = (\bar{R}^T)^{-1}v$ and $b = \sqrt{a - w^Tw}$.

The factorization (4.10) is possible as long as B is positive semidefinite. This follows immediately from its definition, since when J is nonsingular the matrix $B = (JW)^TJW$ is nonsingular if and only if the columns of JW are linearly independent, or equivalently if and only if the columns of W are linearly independent. This will be the case if and only if $\delta^{(0)}$ does not belong to K_m. In the situation where $\delta^{(0)}$ does belong to K_m, $\hat{g}(\bar{y})$ has an infinite number of global minimizers, and B is singular.

To handle the problem of an ill-conditioned or singular B, we use a technique similar to that done in the full N-dimensional quadratic approximation. If C is singular, or its condition number is greater than $1/\sqrt{\gamma}$, where γ is the machine epsilon, then we perturb the quadratic model $\hat{g}(\bar{y})$ to

$$\begin{aligned} h(\bar{y}) &= \hat{g}(\bar{y}) + \frac{1}{2}\mu\bar{y}^T\bar{y} \\ &= \frac{1}{2}F^TF + F^T(JW\bar{y}) + \frac{1}{2}\bar{y}^T(B + \mu I)\bar{y} \\ &= \frac{1}{2}F^TF - d^T\bar{y} + \frac{1}{2}\bar{y}^T(B + \mu I)\bar{y}, \end{aligned}$$

where $\mu = \sqrt{N\cdot\gamma}\,\|B\|_1$. The condition number of $B + \mu I$ is roughly $1/\sqrt{\gamma}$ (see p. 151 in [7] for a justification of this fact). Lemma 4.1 implies that the Newton step to the global minimizer of $h(\bar{y})$, $s = -(B + \mu I)^{-1}d$, solves the minimization problem

$$\min_{\|\bar{y}\|_2 \le \tau} \hat{h}(\bar{y}), \quad \bar{y} \in \mathbf{R}^{m+1},$$

for some $\tau > 0$. It follows easily that this step will also be a descent direction for $f(u)$.

The above procedure is similar to choosing the approximate Newton step s by making it the minimum 2-norm solution of

$$\min_{\bar{y} \in \mathbf{R}^{m+1}} \|F + JW\bar{y}\|_2. \tag{4.11}$$

To see this, let $(JW)^+$ be the pseudo-inverse of JW. Then the solution of (4.11) is $s_{LS} = -(JW)^+F$. For μ small, the step $\hat{s} = -(B + \mu I)^{-1}(JW)^T F$ is similar to s_{LS} in that $\hat{s} \to s_{LS}$ as $\mu \to 0$, and both are orthogonal to the null space of JW for any $\mu > 0$. Again, see [7] for more details.

5. Scaling and preconditioning. The linear Krylov methods discussed in §2 need to be enhanced in order to improve their efficiency and robustness. Two ways of accomplishing this is by using scaling and preconditioning. Scaling is also of particular importance in solving systems of nonlinear equations, as Dennis and Schnabel note [7]. In this section we first discuss scaling of the nonlinear system, its effect on the linear systems to be solved, and then the use of preconditioning in solving the linear systems.

We will allow scaling of both the nonlinear function F and the unknown u in our algorithms. This will be accomplished by using two diagonal scaling matrices D_F and D_u with positive diagonal entries, where the scaled version of (1.1) is

$$\tilde{F}(\tilde{u}) = D_F F(D_u^{-1}\tilde{u}) = 0. \tag{5.1}$$

Here, $\tilde{F} = D_F F$ and $\tilde{u} = D_u u$ are the scaled versions of F and u. The resulting scaled Jacobian $\tilde{J}$ of $\tilde{F}$ is

$$\tilde{J}(\tilde{u}) = \tilde{F}'(\tilde{u}) = D_F J(D_u^{-1}\tilde{u})D_u^{-1}.$$

The scaled Newton equations are then

$$\tilde{J}\tilde{\delta} = -\tilde{F},$$

with $\tilde{\delta} = D_u\delta$. Since we will primarily be using finite-difference versions of the Krylov algorithms, the matrix J will not be available, and hence an explicit scaling of it cannot be performed. However, we will effectively perform an explicit scaling of the Newton equations without ever scaling J. For details of implementing scaling in this way using Krylov methods, see Brown and Hindmarsh [3].

The global strategies discussed in the previous two sections will work with the scaled problem (5.1). However, the scaling will be implemented in an implicit way as follows. Only the diagonal entries of D_F and D_u will be stored, and whenever one of the norms $\|\tilde{F}\|_2$ or $\|\tilde{u}\|_2$ is needed, a call will be made to a routine which computes the scaled norm. The scaled function $\tilde{f} = \frac{1}{2}\tilde{F}^T\tilde{F}$ will be the one used in determining the step from the current iterate to the next.

Without some form of preconditioning, the Krylov methods discussed in §2 are of limited use. Generally, by preconditioning here we mean applying the Krylov method to the equivalent system

$$(P_1^{-1}JP_2^{-1})(P_2\delta) = P_1^{-1}b \text{ or } \bar{J}\bar{\delta} = \bar{b}, \tag{5.2}$$

where $b = -F$. The matrices P_1 and P_2 are chosen in advance so that the preconditioned problem will be easier to solve than the original one. Since linear systems of

the form $P_1 x = c$ and $P_2 x = c$ need to be solved, it is necessary that these additional linear systems be much easier to solve than the original problem. Also, preconditioning makes sense only if the convergence rate of the Krylov method applied to (5.2) is much better than that for the unpreconditioned problem.

The linear system (5.2) is said to be preconditioned on both sides, with P_1 and P_2 referred to as the left and right preconditioners, respectively. If one of these is the identity, then the system is said to be preconditioned on the left or right only.

Incorporating both scaling and preconditioning into the linear system, we then have the scaled preconditioned system

$$(D_F \bar{J} D_u^{-1})(D_u \bar{\delta}) = (D_F \bar{b}) \text{ or } \tilde{\bar{J}}\tilde{\bar{\delta}} = \tilde{\bar{b}}. \tag{5.3}$$

In our case, however, we will need to restrict P_1 to be the identity matrix. The norm of the residual associated with (5.3) is

$$\|\tilde{\bar{r}}\|_2 = \|\tilde{\bar{b}} - \tilde{\bar{J}}\tilde{\bar{\delta}}\|_2,$$

and for general P_1 this is not directly related to the quantity $\|\tilde{r}\|_2 = \|\tilde{b} - \tilde{J}\tilde{\delta}\|_2$ in which we are really interested. Furthermore, our global strategy is based upon the function $\tilde{f} = \frac{1}{2}\tilde{F}^T\tilde{F}$. Hence, the equations (3.3), (3.4) and the similar result used in the dogleg strategy will only hold when P_1 is the identity matrix. Letting $P_1 = I$ and $P_2 = P$, our scaled preconditioned linear system is then

$$(D_F J P^{-1} D_u^{-1})(D_u P \delta) = D_F b. \tag{5.4}$$

The scaled preconditioned Arnoldi and GMRES algorithms are then as follows.

Algorithm: SP–Arnoldi (SP–GMRES)

(1) *Arnoldi process:*
- Choose a tolerance ϵ.
- For an initial guess $\delta^{(0)}$, form $r^{(0)} = -F - J\delta^{(0)}$, where $F = F(u)$ and $J = J(u)$.
- Form $\tilde{r}^{(0)} = D_F r^{(0)}$. Compute $\tilde{\beta} = \|\tilde{r}^{(0)}\|_2$ and $\tilde{v}_1 = \tilde{r}^{(0)}/\tilde{\beta}$.
- For $j = 1, 2, \cdots$, do:
  (a) Compute $\tilde{J}\tilde{v}_j = D_F(J(P^{-1}(D_u^{-1}\tilde{v}_j)))$.
  (b) $\tilde{h}_{i,j} = (\tilde{J}\tilde{v}_j, \tilde{v}_i), \quad i = 1, 2, \cdots, j$.
  (c) $\tilde{\hat{v}}_{j+1} = \tilde{J}\tilde{v}_j - \sum_{i=1}^{j} \tilde{h}_{i,j}\tilde{v}_i$.
  (d) $\tilde{h}_{j+1,j} = \|\tilde{\hat{v}}_{j+1}\|_2$, and
  (e) $\tilde{v}_{j+1} = \tilde{\hat{v}}_{j+1}/\tilde{h}_{j+1,j}$.
  (f) Compute the residual norm $\tilde{\rho}_j = \|\tilde{F} + \tilde{J}\tilde{\delta}^{(j)}\|_2$, of the solution $\tilde{\delta}^{(j)}$ that would be obtained if we stopped at this step.
  (g) If $\tilde{\rho}_j \leq \epsilon$ set $m = j$ and go to (2).

(2) *Form the approximate solution:*

Arnoldi: Define $\tilde{H}_m$ to be the $m \times m$ (Hessenberg) matrix whose nonzero entries are the coefficients $\tilde{h}_{ij}$, $1 \leq i \leq j$, $1 \leq j \leq m$ and $\tilde{V}_m \equiv [\tilde{v}_1, \cdots, \tilde{v}_m]$.
- Find the vector $\tilde{y}_m$ which solves the linear system $\tilde{H}_m \tilde{y} = \tilde{\beta} e_1$.
- Compute $\delta^{(m)} = \delta^{(0)} + P^{-1} D_u^{-1} \tilde{z}^{(m)}$, where $\tilde{z}^{(m)} = \tilde{V}_m \tilde{y}_m$.

GMRES: Define $\tilde{\bar{H}}_m$ to be the $(m+1) \times m$ (Hessenberg) matrix whose nonzero entries are the coefficients $\tilde{h}_{ij}$, $1 \leq i \leq j+1$, $1 \leq j \leq m$ and $\tilde{V}_m \equiv [\tilde{v}_1, \cdots, \tilde{v}_m]$.

- Find the vector $\tilde{y}_m$ which minimizes $\|\tilde{\beta}e_1 - \tilde{\bar{H}}_m\tilde{y}\|_2$ over all vectors $\tilde{y}$ in $\mathbf{R}^m$.
- Compute $\delta^{(m)} = \delta^{(0)} + P^{-1}D_u^{-1}\tilde{z}^{(m)}$, where $\tilde{z}^{(m)} = \tilde{V}_m\tilde{y}_m$.

In this formulation, we have elected to precondition first and scale second. Alternatively, one could do these in the opposite order. The parentheses in the calculation of $\tilde{J}\tilde{v}_j$ indicate that this product is found by first forming $D_u^{-1}\tilde{v}_j$, then solving the system $Pw = D_u^{-1}\tilde{v}_j$, multiplying w by J, and then finally multiplying the result by D_F. Recall that the matrix J here is never formed, and so each operation must be done separately.

6. Numerical testing. In this section, we discuss the relevant implementation details of the methods developed in the previous sections, and then present some results of testing the methods on several problems. We have implemented both global strategies in one solver, the format of which was based upon the overall design of such a package presented in Dennis and Schnabel [7].

6.1. An experimental solver. A solver package called NKSOL (Nonlinear Krylov SOLver for nonlinear systems of equations) has been developed which implements the above methods. NKSOL allows the user to select from among three basic options:

- Arnoldi/IOM as the linear Krylov method with the linesearch backtracking global strategy,
- GMRES/IGMRES as the linear Krylov method with the linesearch backtracking global strategy, and
- GMRES as the linear Krylov method with the dogleg global strategy.

The driver routine, NKSOL, checks for valid input, handles the initial startup of the iteration, and calls a routine NKSTOP to decide when to stop the iteration. It also calls routines MODEL, LNSRCH, and DOGDRV, which perform the following functions. MODEL calls SLVS, which in turn calls either SPIOM or SPIGMR to solve the Newton equations approximately using Arnoldi or GMRES, respectively. LNSRCH determines an acceptable step in the direction provided by MODEL using the linesearch backtracking strategy of §3. DOGDRV is the dogleg strategy driver. It calls DOGSTP, which computes the point on the dogleg curve corresponding to the current trust region size, and TRGUPD, which determines if the step provided by DOGSTP is acceptable and adjusts the trust region size accordingly. See Fig. 6.1.

The user must supply a routine for calculating $F(u)$, and may optionally supply any or all of three additional routines. By default, finite-differences are used to calculate Jv. However, the user may also supply a routine JAC to perform this multiplication. The other user-supplied routines are PSET and PSOL which involve the preconditioner matrix P. Routine PSOL solves the linear system $Pw = c$, and PSET is called once per Newton iterate to set up any matrix data associated with P.

One detail of particular importance is the selection of σ in the difference quotient approximation (2.11). When using finite differences to form an approximate Jacobian matrix in the context of Newton's method, Dennis and Schnabel [7] suggest a stepsize of the form

$$h_j = \sqrt{\eta}\,\max\{|u_j|, \text{typ}u_j\}\,\text{sign}(u_j), \tag{6.1}$$

in the difference quotient

$$\frac{F(u + h_j e_j) - F(u)}{h_j}.$$

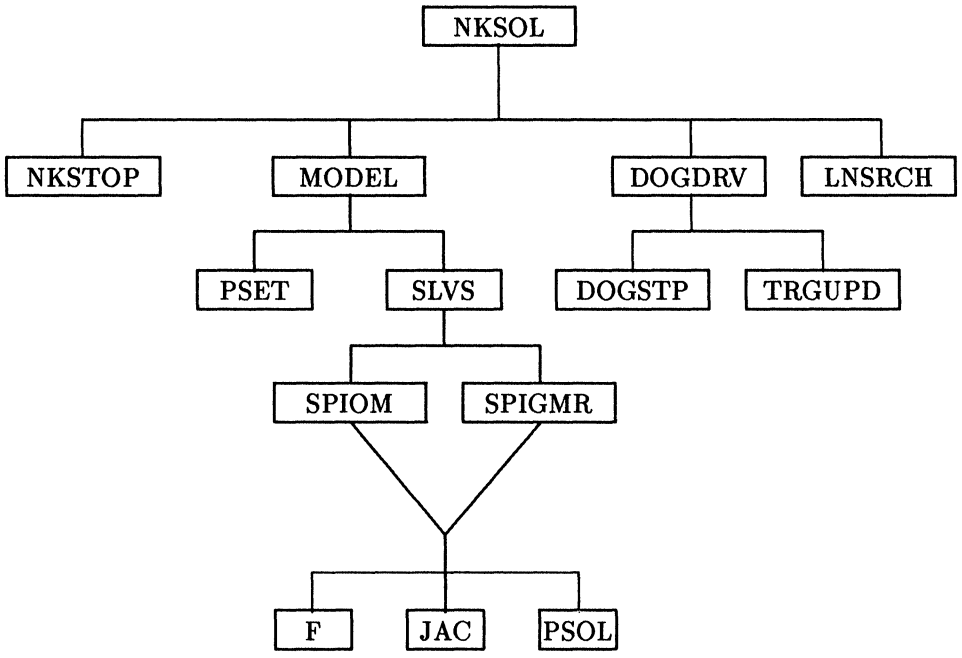

FIG. 6.1. *Simplified Block Structure of NKSOL.*

Here, η represents the relative error in computing $F(u)$, e_j is the jth standard basis vector in $\mathbf{R}^N$, and $\mathrm{typ}u_j > 0$ is a typical size of u_j provided by the user. The size of η depends upon how much work is needed to calculate $F(u)$. In the absence of any information, a value of η equal to machine epsilon is appropriate. The above formula gives an approximation for the jth column of the Jacobian matrix $J(u)$.

In our setting we need only approximate the operation Jv, for a given vector v, and not J itself. To modify (6.1), first note that it can be rewritten as

$$h_j = \sqrt{\eta}\, \max\{|u^T e_j|, \mathrm{typ}u^T e_j\}\, \mathrm{sign}(u^T e_j),$$

where $\mathrm{typ}u = [\mathrm{typ}u_1, \cdots, \mathrm{typ}u_N]^T$. Note also that $\|e_j\|_2 = 1$. If we ignore scaling for the moment, and for a given vector v let $w = v/\|v\|_2$, then an analogous choice for $\bar{\sigma}$ in the difference quotient

$$Jw \approx \frac{F(u + \bar{\sigma} w) - F(u)}{\bar{\sigma}}$$

is

$$\begin{aligned} \bar{\sigma} &= \sqrt{\eta}\, \max\{|u^T w|, \mathrm{typ}u^T |w|\}\, \mathrm{sign}(u^T w) \\ &= \frac{\sqrt{\eta}}{\|v\|_2}\, \max\{|u^T v|, \mathrm{typ}u^T |v|\}\, \mathrm{sign}(u^T v), \end{aligned}$$

where $|w| = [|w_1|, \cdots, |w_N|]^T$, with a similar definition for $|v|$. Choosing $\sigma = \bar{\sigma}/\|v\|_2$ gives us an appropriate value to use in (2.11). If we include scaling, then $\|\cdot\|_2$ and its associated dot product should be replaced by the scaled norm $\|D_u \cdot\|_2$ and its dot

product $(D_u \cdot, D_u \cdot)$. Hence, for a given v, letting $w = v/\|D_u v\|_2$ leads to

$$\begin{aligned} \bar{\sigma} &= \sqrt{\eta}\, \max\{|(D_u u)^T D_u w|, (D_u \mathrm{typ} u)^T D_u |w|\}\, \mathrm{sign}((D_u u)^T D_u w) \\ &= \frac{\sqrt{\eta}}{\|D_u v\|_2} \max\{|(D_u u)^T D_u v|, (D_u \mathrm{typ} u)^T D_u |v|\}\, \mathrm{sign}((D_u u)^T D_u v), \end{aligned}$$

and $\sigma = \bar{\sigma}/\|D_u v\|_2$.

The entries of the diagonal scaling matrix D_u are chosen so as to make the components of the scaled vector $\tilde{u} = D_u u$ all roughly equal in magnitude. Hence, if $D_u = \mathrm{diag}\{d_{11}, \cdots, d_{NN}\}$, then one should use $d_{jj} = \mathrm{typ} u_j^{-1}$ for each j. A similar choice for the entries of the D_F matrix can be made.

The stopping criteria used are those described in Chapter 7 of [7], and are repeated here for convenience. The first test determines whether u_n solves the problem (1.1), i.e., whether $F(u_n) \approx 0$. This is accomplished by using

$$\|D_F F(u_n)\|_\infty \leq \mathrm{FTOL}, \tag{6.2}$$

where a typical value for FTOL is around 10^{-5}. The next test determines whether the algorithm has converged or stalled at u_n. It is of the form

$$\|\mathrm{rel} u\|_\infty \leq \mathrm{STPTOL}, \tag{6.3}$$

where rel$u$ is a vector of length N, which measures the relative change in u from one step to the next. Its components are defined by

$$\mathrm{rel} u_j = \frac{|(u_{n+1})_j - (u_n)_j|}{\max\{|(u_{n+1})_j|, \mathrm{typ} u_j\}}$$

for $j = 1, \cdots, N$. If t significant digits are desired in the solution, then STPTOL should be set to 10^{-t}. (Alternatively, the user can supply his own routines for calculating the size of $F(u_n)$ and the relative change in u that appears in (6.2) and (6.3).) A maximum steplength STPMX is imposed in both the linesearch and dogleg strategies, and if five consecutive steps are this long, then the algorithm is terminated. There is also a maximum number ITMAX of iterations that can be performed on any one call to NKSOL. Currently, this value is 200, but it can optionally be set by the user.

In the test results of the remainder of this section, the following counters and a return flag will be helpful. These are defined as follows:

NFE	–	number of F evaluations
NNI	–	number of nonlinear iterations
NLI	–	number of linear iterations within the Krylov method
NB	–	number of backtracks within the linesearch algorithm, and number of extra F evaluations used by the dogleg strategy
NCFL	–	number of times that the linear solver failed to reduce the residual norm by a factor η_n in $m_{\max}$ steps
ITERM	–	termination flag (=1 if (6.2) holds, =2 if (6.3) holds, and =3 if (3.10) fails to hold).

For ϵ_n we use $\epsilon_n = \eta_n \|F\|_2$, where $\eta_n = (\frac{1}{2})^n$ for $n = 1, 2, \cdots$, and $F = F(u_n)$. A default value of $m_{\max} = 10$ is used with the full orthogonalization version of each Krylov method. If $m = m_{\max}$, but $\|r^{(m)}\|_2 > \eta_n \|F\|_2$, we simply go ahead and use the last computed GMRES or Arnoldi step. The counter NCFL equals the number of Newton iteration steps for which this happened.

TABLE 6.1
Results for the Bratu problem, $\lambda = 1$.

	No Preconditioning			Laplacian Preconditioning		
	MF = 1	MF = 2	MF = 3	MF =1	MF = 2	MF = 3
NFE	151	205	150	28	28	27
NNI	15	20	15	6	6	6
NLI	134	184	134	20	21	20
NCFL	13	18	13	0	0	0
NB	1	0	0	1	0	0
ITERM	1	1	1	1	1	1

6.2. Test problem 1. Solution of a Bratu problem. As a first test problem we chose to solve the nonlinear partial differential equation

$$-\Delta u + \alpha u_x + \lambda e^u = f \tag{6.4}$$

over the unit square of $\mathbf{R}^2$ with Dirichlet boundary conditions. This is a standard problem, a simplified form of which is known as the Bratu problem [13]. After discretization by 5-point finite differencing, we obtain a large system of nonlinear equations of size N, where $N = {n_x}^2$ and n_x is the number of meshpoints in each direction. The right-hand side f is chosen so that the solution of the discretized problem is known to be the constant unity. As a result, the equation will always have at least one solution. In fact, it is known that for $\lambda \geq 0$ there is always a unique solution to the problem (see [13]). In this test we took $n_x = 32$, yielding a nonlinear system of $N = 1024$ unknowns, and $\alpha = 10.0, \lambda = 1.0$. It is possible to precondition this problem by the Laplacian operator. To this end we can use a fast Poisson solver such as the subroutine HWSCRT from FISHPACK [26]. We tested the three basic methods from NKSOL, each of them with and then without this preconditioning. The three methods correspond to the following method flags:

MF = 1 – The dogleg technique using GMRES as a linear solver.
MF = 2 – The backtracking linesearch technique using Arnoldi's method as a linear solver.
MF = 3 – The backtracking linesearch technique using the GMRES method as a linear solver.

For Arnoldi and GMRES the default value of $m_{\max} = 10$ was used, and the values of FTOL and STPTOL were 10^{-7} and 10^{-10}, respectively. No scaling was used and the initial guess was always taken to be zero. Table 6.1 shows the results for all three methods with and without preconditioning, while the graph of Fig. 6.2 plots the value of $f = \frac{1}{2}F^TF$ on a logarithmic scale in powers of 10 as a function of NFE, the number of function evaluations consumed to reach that value of f. Here, the performances of the three basic methods MF=1,2,3 differ very little as is shown in Table 6.1 and in the plot of Fig. 6.2, with the exception that MF=1 and 3 have a slight edge over MF=2. On the other hand, preconditioning was extremely helpful. Note that both the number of nonlinear iterations and the total number of linear steps is much reduced by the use of the preconditioner. We should mention that for this problem it is possible to take advantage of other subroutines from FISHPACK. For example, instead of preconditioning with the Laplacian we could have used BLCKTRI to precondition by the discrete version of the partial differential operator $\Delta + \partial/\partial x$ or even of $\Delta + \kappa + \partial/\partial x$ where κ could be, for example, some average of $\lambda e^{u_{n-1}}$ over the domain.

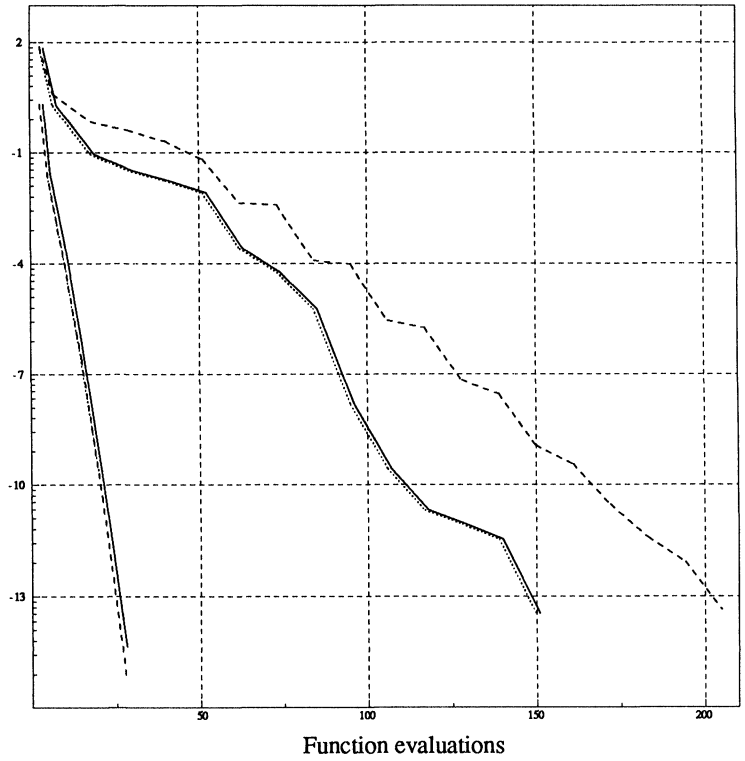

FIG. 6.2. *Results for the Bratu problem, $\lambda = 1$. Solid line: $MF = 1$; dashed line: $MF = 2$; dotted line: $MF = 3$. Lower curves: with preconditioning; upper curves: no preconditioning.*

In our second experiment with this test problem, we took a negative value for λ to make the problem more difficult to solve. When $\lambda < 0$ the Jacobians in Newton's method may not be positive real and this is a source of difficulty for the linear system solvers. With $\lambda = -5$ we obtain the results of Table 6.2 and the plot of Fig. 6.3. We observe again that there is little difference between the three methods, again with the exception that MF=1 and 3 appear to have a slight edge over MF=2 when there is no preconditioning.

6.3. Test problem 2. The driven cavity problem. This second test problem is the classical *driven cavity* problem from incompressible fluid flow. In stream function–vorticity formulation the equations are

$$\nu\Delta\omega + (\psi_{x_2}\omega_{x_1} - \psi_{x_1}\omega_{x_2}) = 0 \text{ in } \Omega, \tag{6.5}$$

$$-\Delta\psi = \omega \text{ in } \Omega, \tag{6.6}$$

$$\psi = 0 \text{ on } \partial\Omega, \tag{6.7}$$

$$\frac{\partial\psi}{\partial n}(x_1, x_2)|_{\partial\Omega} = \begin{cases} 1 & \text{if } x_2 = 1, \\ 0 & \text{if } 0 \le x_2 < 1. \end{cases} \tag{6.8}$$

Here, $\Omega = \{(x_1, x_2) : 0 < x_1 < 1, 0 < x_2 < 1\}$, and the viscosity ν is the reciprocal of the Reynolds number Re. In terms of ψ alone, (6.5) and (6.6) are replaced by

$$\nu\Delta^2\psi + (\psi_{x_2}(\Delta\psi)_{x_1} - \psi_{x_1}(\Delta\psi)_{x_2}) = 0 \text{ in } \Omega, \tag{6.9}$$

subject to the boundary conditions (6.7) and (6.8). For more details on the problem definition see [13].

TABLE 6.2
Results for the Bratu problem, $\lambda = -5$.

	No Preconditioning			Laplacian Preconditioning		
	MF = 1	MF = 2	MF = 3	MF =1	MF = 2	MF = 3
NFE	195	230	216	30	29	29
NNI	19	22	21	6	6	6
NLI	174	204	194	22	22	22
NCFL	17	20	19	0	0	0
NB	1	3	0	1	0	0
ITERM	1	1	1	1	1	1

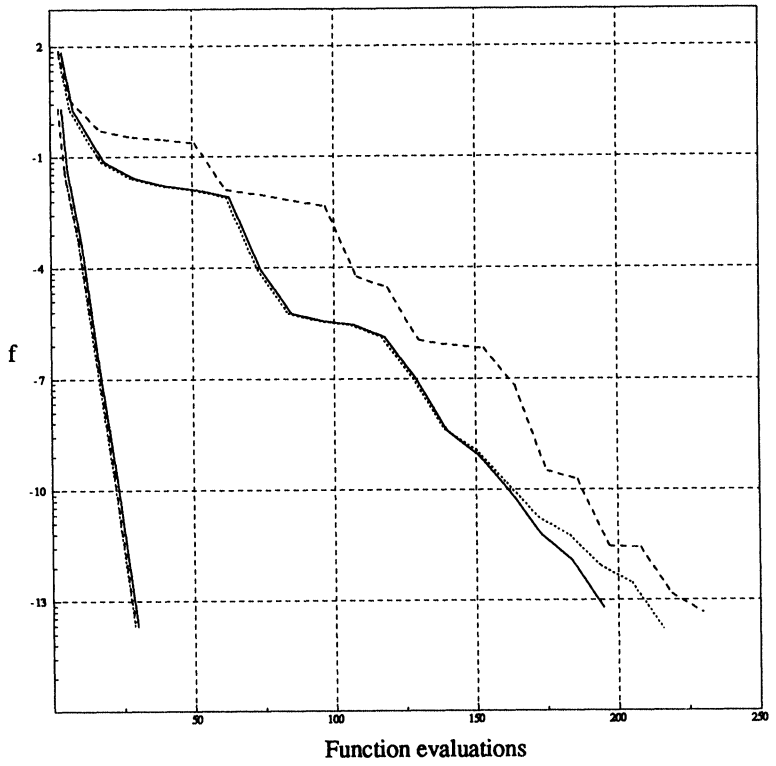

FIG. 6.3. *Results for the Bratu problem, $\lambda = -5$. Solid line: $MF = 1$; dashed line: $MF = 2$; dotted line: $MF = 3$. Lower curves: with preconditioning; upper curves: no preconditioning.*

TABLE 6.3
Results for the driven cavity problem ($Re = 500$).

BIHAR	Cholesky Decomposition		
	MF = 1	MF = 2	MF = 3
NFE	146	308	145
NNI	14	27	14
NLI	130	261	130
NCFL	11	26	11
NB	1	19	0
ITERM	1	1	1

TABLE 6.4
Results for the driven cavity problem ($Re = 1500$ *and* 2000).

	Re = 1500			Re = 2000	
	MF = 1	MF = 2	MF = 3	MF = 1	MF = 3
NFE	484	393	472	552	567
NNI	31	16	30	35	36
NLI	451	226	436	512	527
NCFL	30	15	29	34	35
NB	1	150	5	4	3
ITERM	1	3	1	1	1

Equation (6.9) was discretized using piecewise linear finite elements in a manner similar to that for the biharmonic problem $\Delta^2\psi = f$ outlined in §3 of [14]. This discretization is also equivalent to that obtained using standard finite differences. The resulting nonlinear system has $N = 63^2 = 3969$ unknowns. For a preconditioner, we use the discretized version of $P = \nu\Delta^2$. For a uniform mesh, systems of the form $Pw = c$ can be solved very efficiently using a fast solver, for example BIHAR developed by Bjørstad [1]. This preconditioner should be very effective for small Reynolds numbers, with decreasing effectiveness as the Reynolds number grows. For comparison, we considered $Re = 500$, 1500, 2000, 3000, and 5000. All runs were made on a CRAY-1 at LLNL, with FTOL = 10^{-7}, STPTOL = 10^{-8}, and the scaling matrices D_F and D_u both equal to the identity (i.e., no scaling). The initial guess ψ_0 was simply the zero vector.

The results for $Re = 500$ are given in Table 6.3. We used the Cholesky decomposition option in BIHAR for the solution of the biharmonic problem. This option computes an exact solution with a minimal storage requirement. For this Reynolds number, the preconditioner is very effective and the two GMRES method options perform similarly, whereas the Arnoldi/Linesearch option takes more nonlinear iterations but still achieves convergence. See Fig. 6.4 for a plot of function evaluations versus values of $f = \frac{1}{2}F^TF$, and Fig. 6.5 for a contour plot of the stream function values in this case. The contour levels plotted are

$$\begin{aligned}\psi \;=\; & -0.1, -0.08, -0.06, -0.04, -0.02, 0.0, \\ & 0.0025, 0.001, 0.0005, 0.0001, 0.00005.\end{aligned}$$

The results for $Re = 1500$ and 2000 are given in Table 6.4. For these higher Reynolds numbers the preconditioner is less effective, and the original runs (using

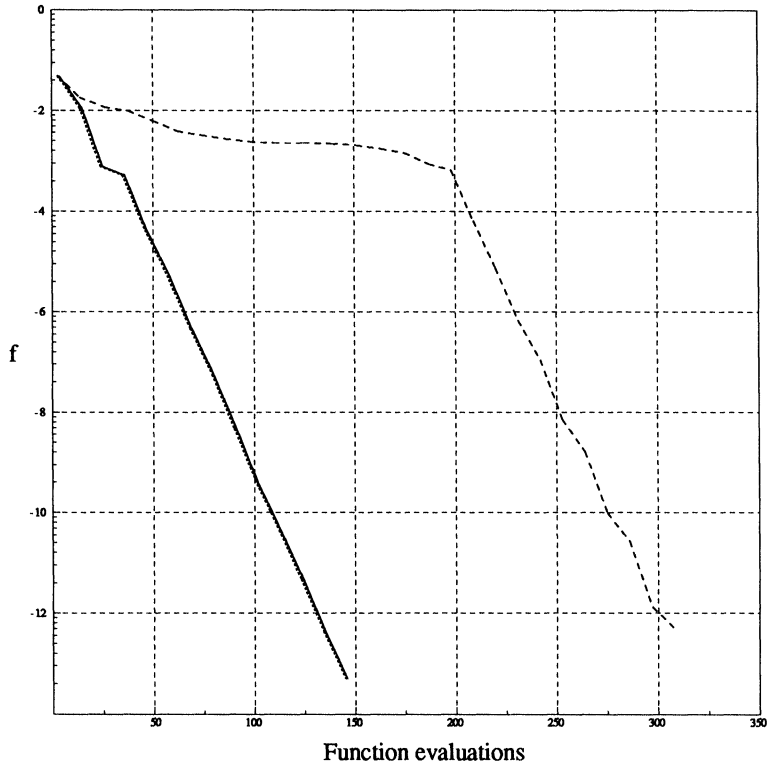

FIG. 6.4. *Reynolds number* 500. *Solid line*: $MF = 1$; *dashed line*: $MF = 2$; *dotted line*: $MF = 3$. *Cholesky preconditioning.*

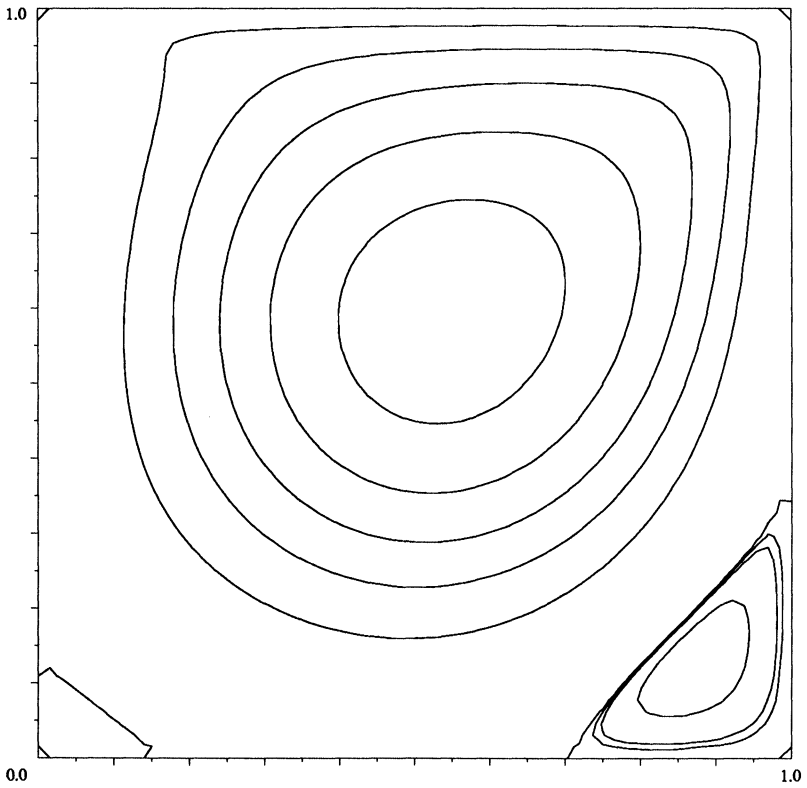

FIG. 6.5. *Reynolds number* 500.

TABLE 6.5
Results for the driven cavity problem (Re = 3000 and 5000).

Re	3000	5000
	MF = 1	MF = 1
NFE	904	1407
NNI	57	88
NLI	845	1315
NCFL	56	87
NB	1	3
ITERM	1	1

the default value of $m_{\max} = 10$) failed to converge for all MF values. Increasing $m_{\max}$ to 15 yielded the results in the table. A possible reason for the convergence failures is that since the preconditioner is less effective, the default Krylov subspace dimension is not large enough to generate sufficiently good descent directions for the nonlinear iteration. In addition, the Arnoldi/Linesearch combination performed very poorly, and the high number of backtracks suggest that Arnoldi is producing very poor descent directions (even with the higher $m_{\max}$ value). We say more about this below. See Figs. 6.6 and 6.7 for function evaluations versus f, and Figs. 6.8 and 6.9 for contour plots. Table 6.5 contains the results for Reynolds numbers 3000 and 5000. We elected to test only the dogleg strategy for these higher Reynolds numbers. While GMRES is having difficulty solving the linear systems for high Reynolds numbers the nonlinear iteration is still achieving convergence. (See Figs. 6.10–6.13.)

Finally, we note that the Arnoldi/Linesearch option in general performs very poorly for the large Reynolds number cases. This may be partly due to the fact that the Jacobian matrix $J(\psi)$ becomes nearly skew-symmetric for small ν. While it is not surprising that both Arnoldi and GMRES may perform poorly in this case, the directions each computes can be vastly different. It follows from (3.3) and (3.4) that the derivative $\nabla f^T \delta$ is independent of the residual associated with the Arnoldi step $\delta = \delta_A$, whereas this is not the case for the GMRES step $\delta = \delta_{GM}$. If we use the norm of the residual as a measure of how good an approximation δ is to the full Newton step, then it appears that the GMRES step δ_{GM} may in general be preferable. In other words, a poor GMRES step is in general better than a poor Arnoldi step.

7. Conclusion. Krylov subspace methods for linear systems can be combined with Newton iteration and globally convergent strategies to obtain effective algorithms for general nonlinear systems of equations. We have shown how the Arnoldi and GMRES methods can be fitted into such a combination and have discussed some of the relevant implementation details. We have also implemented these algorithms in an experimental solver NKSOL. The test problems we have considered show that these methods (with suitable preconditioning) can be quite effective for some classes of problems. Propositions 3.1 and 3.2 and the remarks at the end of the previous section indicate that GMRES may be slightly better than other available Krylov methods, at least in the nonlinear equations setting. Similar to the situation in linear equations, one observes that the choice of a preconditioner is more important than that of the Krylov subspace method. In many instances, nonlinear preconditioners can be built from linear parts of the nonlinear equations as was shown in the examples of § 6. Another way of preconditioning is to transform the initial nonlinear equation, for example, by solving $u - M(u) = 0$ where the operator comes from some known fixed

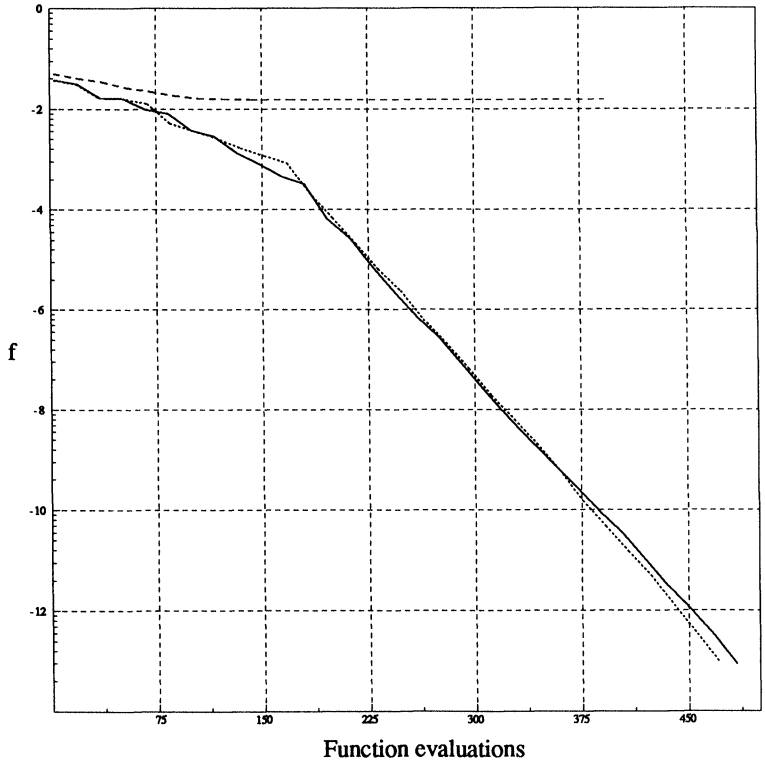

FIG. 6.6. *Reynolds number* 1500. *Solid line*: $MF = 1$; *dashed line*: $MF = 2$; *dotted line*: $MF = 3$. *Cholesky preconditioning.*

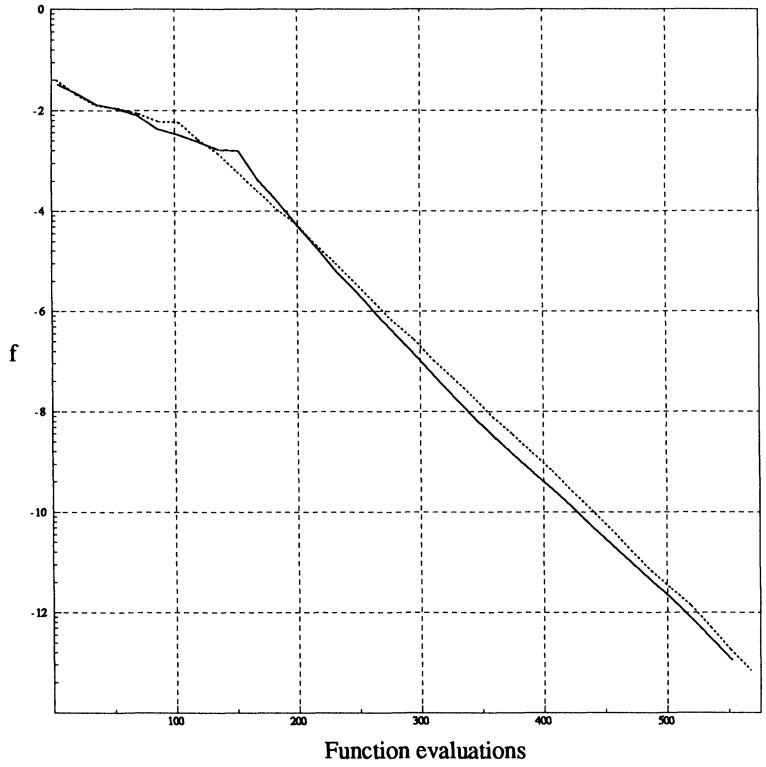

FIG. 6.7. *Reynolds number* 2000. *Solid line*: $MF = 1$; *dotted line*: $MF = 3$. *Cholesky preconditioning.*

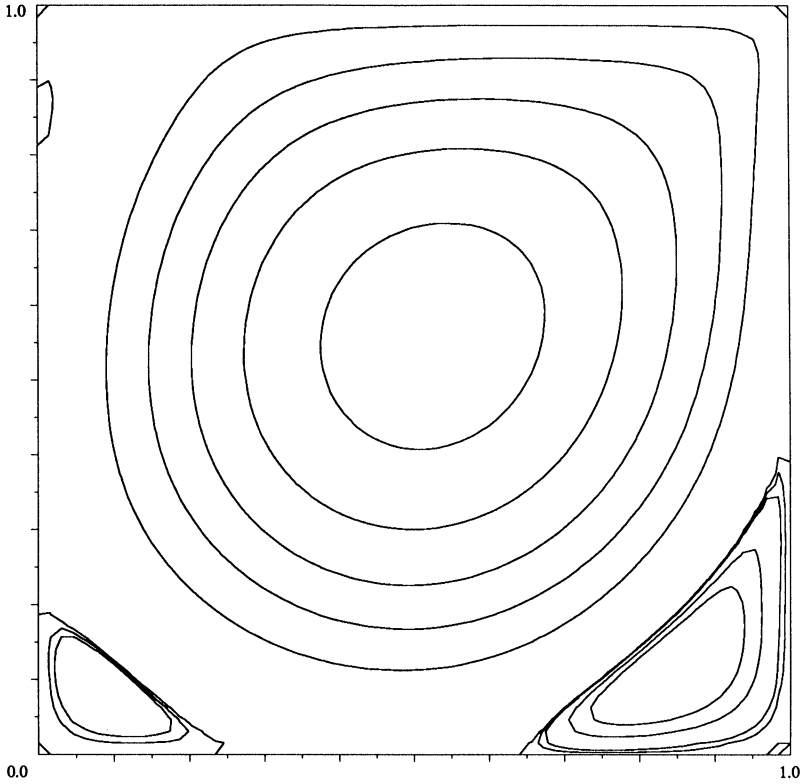

FIG. 6.8. *Reynolds number* 1500.

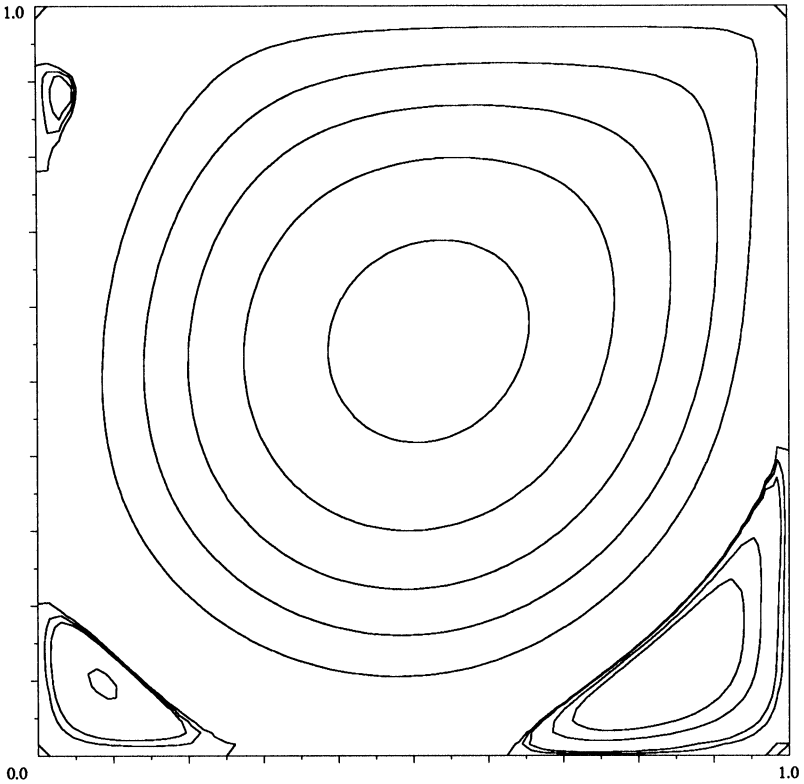

FIG. 6.9. *Reynolds number* 2000.

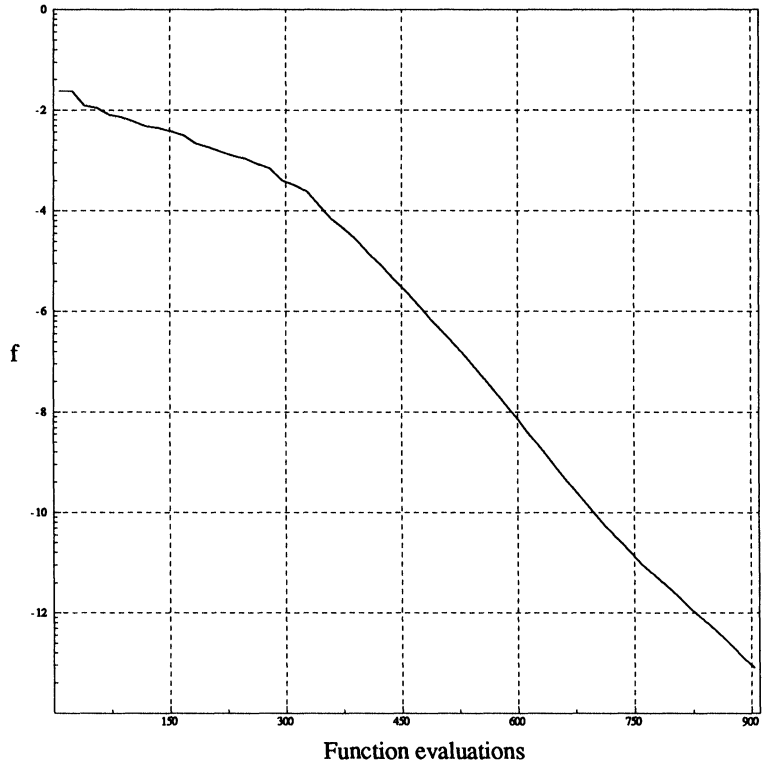

FIG. 6.10. *Reynolds number* 3000. *Solid line*: $MF = 1$. *Cholesky preconditioning.*

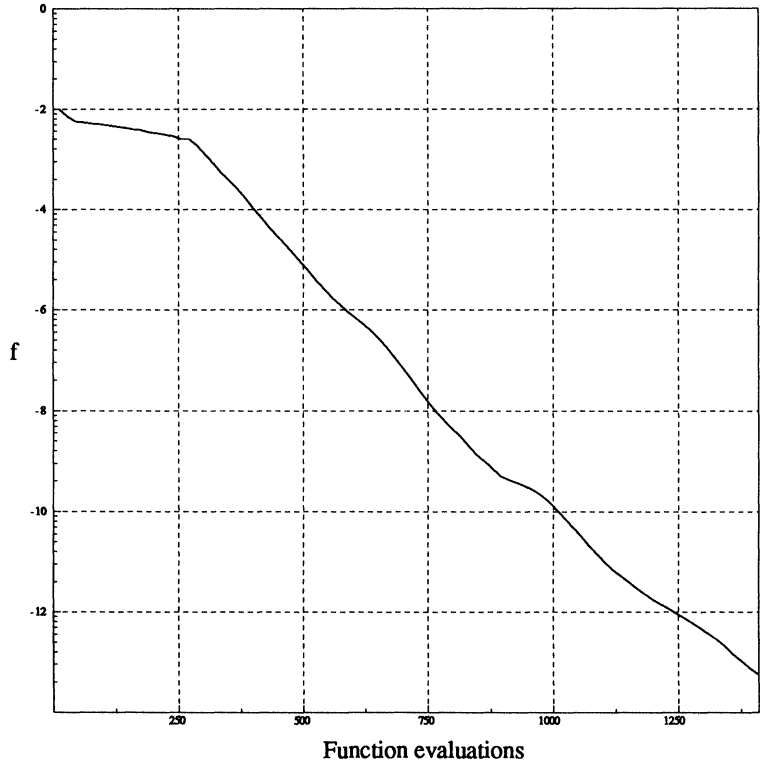

FIG. 6.11. *Reynolds number* 5000. *Solid line*: $MF = 1$. *Cholesky preconditioning.*

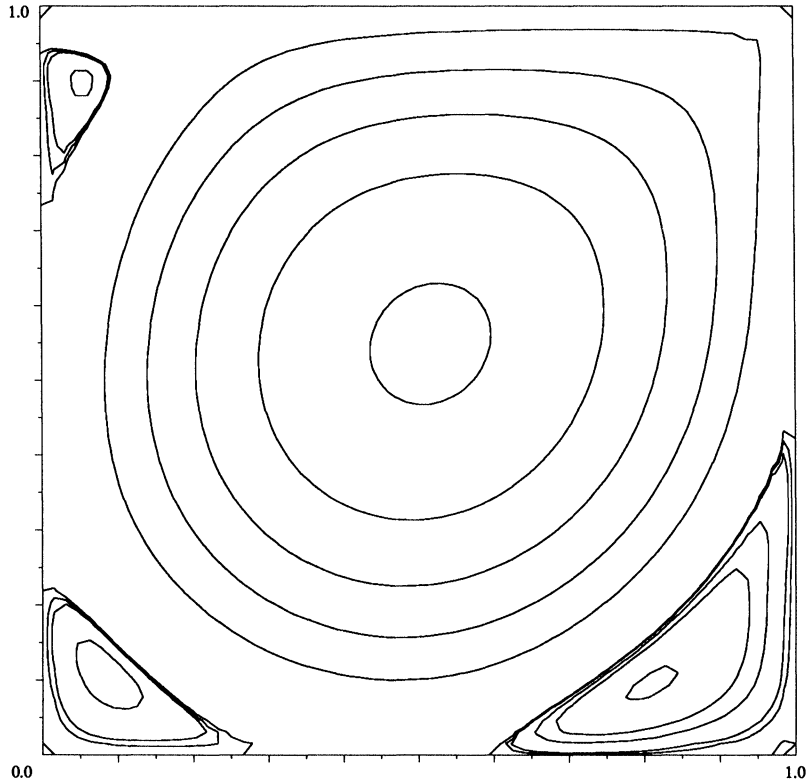

FIG. 6.12. *Reynolds number* 3000.

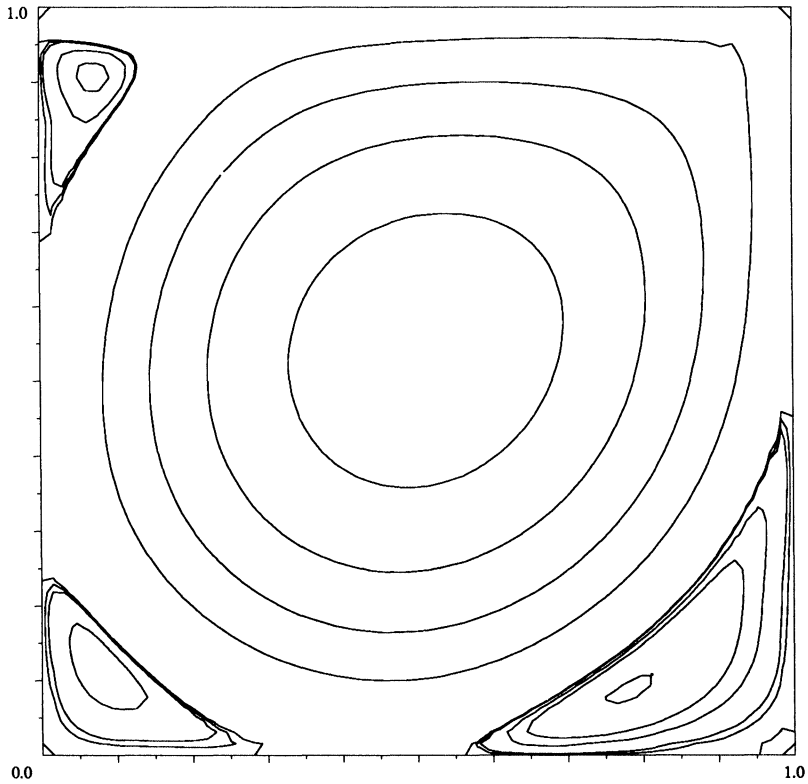

FIG. 6.13. *Reynolds number* 5000.

point iteration $u_{n+1} = M(u_n)$. This opens up many possibilities for using Krylov subspace methods as accelerators of existing codes based on nonlinear stationary iterations. There are many issues that remain to be investigated. Among them is the possibility of exploiting information gathered during the previous nonlinear steps, such as previous Krylov subspaces and the corresponding Jacobian in that space. Another interesting question, related to the fluid dynamics problem of § 6.2, is whether solving the stationary problem directly is more effective than solving the time-dependent problem over a large enough time period. This second approach would in effect constitute a sort of continuation technique with continuation parameter the time t, and it might very well be the case that for very hard problems a combination of the methods described in this paper and a form of continuation will be the most robust approach.

Finally, this paper has focused on presenting a general purpose nonlinear equation solver based on nonlinear Krylov subspace techniques, enhanced by global convergence strategies. The algorithms we have implemented are all based on extrapolations of globally convergent methods for full N-dimensional quadratic models. In future work, we intend to study the convergence properties of these combined global/Newton–Krylov methods.

REFERENCES

[1] P. BJØRSTAD, *Fast numerical solution of the biharmonic Dirichlet problem on rectangles,* SIAM J. Numer. Anal., 20(1983), pp. 59-71.

[2] P. N. BROWN, *A local convergence theory for combined inexact-Newton/finite-difference projection methods,* SIAM J. Numer. Anal., 24(1987), pp. 407-434.

[3] P. N. BROWN AND A. C. HINDMARSH, *Matrix-free methods for stiff systems of* ODEs, SIAM J. Numer. Anal., 24(1987), pp. 610-638.

[4] _____, *Reduced storage methods in stiff* ODE *systems,* J. Appl. Math. Comput., 31(1989), pp. 40-91.

[5] T. F. CHAN AND K. R. JACKSON, *The use of iterative linear equation solvers in codes for large systems of stiff* IVPs *for* ODEs, SIAM J. Sci. Statist. Comput., 7(1986), pp. 378-417.

[6] R. S. DEMBO, S. C. EISENSTAT, AND T. STEIHAUG, *Inexact Newton methods,* SIAM J. Numer. Anal., 19(1982), pp. 400-408.

[7] J. E. DENNIS AND R. B. SCHNABEL, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations,* Prentice-Hall, Englewood Cliffs, NJ, 1983.

[8] H. C. ELMAN, *Iterative methods for large sparse nonsymmetric systems of linear equations,* Ph.D. thesis, Computer Science Department, Yale University, New Haven, CT, 1982.

[9] L. E. ERIKSSON AND A. RIZZI, *Analysis by computer of the convergence of discrete approximations to the Euler equations,* in Proc. 1983 AIAA Conference, Denver, 1983; AIAA paper number 83-1951, Denver, CO, 1983, pp. 407-442.

[10] D. GAY, *Computing optimal locally constrained steps,* SIAM J. Sci. Statist. Comput., 2(1981), pp. 186-197.

[11] W. C. GEAR AND Y. SAAD, *Iterative solution of linear equations in* ODE *codes,* SIAM J. Sci. Statist. Comput., 4(1983), pp. 583-601.

[12] P. E. GILL, W. MURRAY, AND M. WRIGHT, *Practical Optimization,* Academic Press, New York, 1981.

[13] R. GLOWINSKI, H. B. KELLER, AND L. REINHART, *Continuation-conjugate gradient methods for the least squares solution of nonlinear boundary value problems,* SIAM J. Sci. Statist. Comput., 6(1985), pp. 793-832.

[14] R. GLOWINSKI AND O. PIRONNEAU, *Numerical methods for the first biharmonic equation and for the two-dimensional Stokes problem,* SIAM Rev., 21(1979), pp. 167-212.

[15] A. L. HAGEMAN AND D. M. YOUNG, *Applied Iterative Methods,* Academic Press, New York, 1981.

[16] T. KERKHOVEN AND Y. SAAD, *Acceleration techniques for decoupling algorithms in semiconductor simulation,* CSRD Report 684, University of Illinois at Urbana Champaign, Urbana, IL, 1987.

[17] W. L. MIRANKER AND I.-L. CHERN, *Dichotomy and conjugate gradients in the stiff initial value problem,* Linear Algebra Appl., 36(1981), pp. 57-77.
[18] J. J. MORÉ, *The Levenberg–Marquardt algorithm: Implementation and theory,* in Numerical Analysis, G. A. Watson, ed., Lecture Notes in Mathematics 630, Springer-Verlag, Berlin, New York, 1977, pp. 105-116.
[19] J. J. MORÉ, B. S. GARBOW, AND K. E. HILLSTROM, *User guide for* MINPACK-1, Tech. Report ANL-80-74, Argonne National Laboratories, Argonne, IL, 1980.
[20] D. P. O'LEARY, *A discrete Newton algorithm for minimizing a function of many variables,* Math. Programming, 23 (1982), pp. 20-33.
[21] M. J. D. POWELL, *A hybrid method for nonlinear equations,* P. Rabinowitz, ed., Numerical Methods for Nonlinear Equations, Gordon-Breach, New York, 1970.
[22] Y. SAAD, *Krylov subspace methods for solving unsymmetric linear systems,* Math. Comp., 37(1981), pp. 105-126.
[23] Y. SAAD AND M. H. SCHULTZ, *Conjugate gradient-like algorithms for solving nonsymmetric linear systems,* Math. Comp., 44(1985), pp. 417-424.
[24] _____, GMRES: *A generalized minimal residual algorithm for solving nonsymmetric linear systems,* SIAM J. Sci. Statist. Comput., 7(1986), pp. 856-869.
[25] T. STEIHAUG, *The conjugate gradient method and trust regions in large scale optimization,* SIAM J. Numer. Anal., 20 (1983), pp. 626-637.
[26] P. N. SWARTZTRAUBER AND R. A. SWEET, *Algorithm* 541: *Efficient Fortran subprograms for the solution of separable elliptic partial differential equations,* ACM Trans. Math. Software, 5(1979), pp. 352-364.
[27] L. B. WIGTON, D. P. YU, AND N. J. YOUNG, GMRES *acceleration of computational fluid dynamics codes,* in Proc. 1985 AIAA Conference, Denver, CO, 1985.