Computational Experience with Globally Convergent Descent Methods for Large Sparse Systems of Nonlinear Equations

Lukšan, Ladislav
1996

INSTITUTE OF COMPUTER SCIENCE

ACADEMY OF SCIENCES OF THE CZECH REPUBLIC

Computational Experience with Globally Convergent Descent Methods for Large Sparse Systems of Nonlinear Equations

Ladislav Lukšan Jan Vlček

Technical report No. V-668

April 1996

Institute of Computer Science, Academy of Sciences of the Czech Republic
Pod vodárenskou věží 2, 182 07 Prague 8, Czech Republic
phone: (+4202) 66053260 fax: (+4202) 8585789
e-mail: luksan@uivt.cas.cz

INSTITUTE OF COMPUTER SCIENCE

ACADEMY OF SCIENCES OF THE CZECH REPUBLIC

Computational Experience with Globally Convergent Descent Methods for Large Sparse Systems of Nonlinear Equations

Ladislav Lukšan[1] Jan Vlček

Technical report No. V-668
April 1996

Abstract

This paper is devoted to globally convergent Armijo-type descent methods for solving large sparse systems of nonlinear equations. These methods include the discrete Newton method and a broad class of Newton-like methods based on various approximations of the Jacobian matrix. We propose a general theory of global convergence together with a robust algorithm including a special restarting strategy. This algorithm is based on the truncated preconditioned smoothed CGS method for solving nonsymmetric systems of linear equations. After reviewing 12 particular Newton-like methods, we propose results of extensive computational experiements. These results demonstrate high efficiency of the proposed algorithm.

Keywords

Nonlinear equations, Armijo-type descent methods, Newton-like methods, truncated methods, global convergence, nonsymmetric linear systems, conjugate gradient-type methods, residual smoothing, computational experiments

AMS classification. 62J02

[1]This work was supported by the Grant Agency of the Czech Republic under grant 201/96/0918

1 Introduction

Let f be a continuously differentiable mapping from $\mathcal{R}^n$ to $\mathcal{R}^n$ in the form $f(x) = (f_1(x), f_2(x), \ldots, f_n(x))^T$ and consider the system of nonlinear equations

$$f(x) = 0 \tag{1.1}$$

for some unknown point $x \in \mathcal{R}^n$. Let $J(x)$ denote the Jacobian matrix of the mapping f with

$$(J(x))_{ij} = \frac{\partial f_i(x)}{\partial x_j}, 1 \leq i \leq n, 1 \leq j \leq n.$$

Let $x_1 \in \mathcal{R}^n$, $\overline{F} \geq \|f(x_1)\|$ and $\overline{\Delta} \geq 0$. Denote

$$\mathcal{L}(\overline{F}) = \{x \in \mathcal{R}^n : \|f(x)\| \leq \overline{F}\}$$

and

$$\mathcal{D}(\overline{F}, \overline{\Delta}) = \{x \in \mathcal{R}^n : \|x - y\| \leq \overline{\Delta} \text{ for some } y \in \mathcal{L}(\overline{F})\}$$

Throughout the paper we will use Euclidean vector norm and spectral matrix norm respectively and suppose the following assumptions hold:

A1: The Jacobian matrix $J(x)$ is defined and bounded on $\mathcal{D}(\overline{F}, \overline{\Delta})$, i.e.

$$\|J(x)\| \leq \overline{J}, \quad \forall x \in \mathcal{D}(\overline{F}, \overline{\Delta})$$

A2: The Jacobian matrix $J(x)$ is Lipschitz continuous on $\mathcal{D}(\overline{F}, \overline{\Delta})$, i.e.

$$\|J(y) - J(x)\| \leq \overline{L}\|y - x\| \quad \forall x, y \in \mathcal{D}(\overline{F}, \overline{\Delta})$$

In this paper, we will concentrate on a class of Armijo-type descent methods for the solution to the system (1.1), which generate the sequence of points $x_i \in \mathcal{R}^n$, $i \in \mathcal{N}$, such that

$$x_{i+1} = x_i + \alpha_i s_i, \quad i \in \mathcal{N}, \tag{1.2}$$

where $s_i \in \mathcal{R}^n$ is the direction vector determined as an inexact solution of the linear system $A_i s + f_i = 0$ and where the stepsize α_i is selected to guarantee sufficient decrease of $\|f(x)\|$. Here A_i is an approximation of the matrix $J_i = J(x_i)$ and $f_i = f(x_i)$.

For investigating Armijo-type descent methods we also use the objective function

$$F(x) = \frac{1}{2}\|f(x)\|^2, \tag{1.3}$$

which has the same local and global minima as the norm $\|f(x)\|$, and denote $F_i = F(x_i)$, $g_i = g(x_i)$, $i \in \mathcal{N}$, where $g(x) = J^T(x)f(x)$ is the gradient of $F(x)$.

While the influence of inexactness of the solution of the system $A_i s + f_i = 0$ on global convergence was succesfully studied in [6]-[7], [17]-[18], [24], the influence of inexactness

of the approximation A_i of the Jacobian matrix J_i has not been considered, with the exception of the case of finite difference approximation of the Jacobian matrix studied in [5], which deals only with local convergence. Therefore, we consider both of these inexactnesses in this paper.

The paper is organized as follows. In Section 2, we propose a class of Armijo-type descent methods and formulate conditions for their global convergence. These conditions (especially assumption A4) cannot be verified in general, but our theory is useful for particular algorithmic realizations. To globalize Newton-like methods, we propose an implementable algorithm, based on restarts, which does not use assumption A4, while it is still globally convergent (if standard assumptions hold). Furthermore, we give a short description of the preconditioned smoothed CGS algorithm used for direction determination. Section 3 is devoted to the description of various Newton-like methods which can be realized by Algorithm 1. Finally, Section 4 contains results of computational experiments which demonstrate high efficiency of Newton-like methods realized by Algorithm 1 with preconditioned smoothed CGS subalgorithm.

2 Descent methods

We begin with the definition of a class of Armijo-type descent methods for the solution to a system of nonlinear equations. More detailed information can be found in Algorithm 1.

Definition 1 *We say that the basic method $x_{i+1} = x_i + \alpha_i s_i$, $i \in \mathcal{N}$, for solution to a system of nonlinear equations $f(x) = 0$ is an Armijo-type descent method (D), if the following conditions hold:*
D1: Direction vectors $s_i \in \mathcal{R}^n$, $i \in \mathcal{N}$, are determined so that

$$\|A_i s_i + f_i\| \leq \overline{\omega}\|f_i\|, \tag{2.1}$$

where $0 \leq \overline{\omega} < 1$.
D2: Steplengths $\alpha_i > 0$, $i \in \mathcal{N}$, are chosen so that α_i is the first member of the sequence α_i^j, $j \in \mathcal{N}$, where $\alpha_i^1 = 1$ and $\underline{\beta}\alpha_i^j \leq \alpha_i^{j+1} \leq \overline{\beta}\alpha_i^j$ with $0 < \underline{\beta} \leq \overline{\beta} < 1$, satisfying

$$F_{i+1} - F_i \leq -2\underline{\rho}(1 - \overline{\omega})\alpha_i F_i, \tag{2.2}$$

where $0 < \underline{\rho} < 1$.

Condition (2.2) is closely related to the condition

$$\|f_{i+1}\| - \|f_i\| \leq -\underline{\rho}(1 - \overline{\omega})\alpha_i\|f_i\|$$

used in [18].

In subsequent considerations, we frequently use the following assumptions:

A3: Matrices $J_i^{-1} = J^{-1}(x_i)$, $i \in \mathcal{N}$, are defined and uniformly bounded on the sequence of points $x_i \in \mathcal{L}(\overline{F})$, $i \in \mathcal{N}$, generated by the Armijo-type descent method (D), i.e.

$$\|J_i^{-1}\| \leq 1/\underline{J}, \quad \forall i \in \mathcal{N}.$$

A4: A3 holds and a constant $0 \leq \overline{\vartheta} < (1/2)(1-\overline{\omega})\underline{J}$ *exist, such that*

$$\|A_i - J_i\| \leq \overline{\vartheta}, \quad \forall i \in \mathcal{N}.$$

A5: Matrices A_i^{-1}, $i \in \mathcal{N}$, *are defined and uniformly bounded, i.e.*

$$\|A_i^{-1}\| \leq 1/\underline{A}, \quad \forall i \in \mathcal{N}.$$

Lemma 1 *Let assumption A4 be satisfied and let (2.1) hold. Then a constant* $0 \leq \overline{\eta} < 1$ *exists, such that*

$$\|J_i s_i + f_i\| \leq \overline{\eta}\|f_i\|, \quad \forall i \in \mathcal{N}. \tag{2.3}$$

Proof Since A4 holds, we can write

$$\|A_i s_i\| \geq \|J_i s_i\| - \|(A_i - J_i)s_i\| \geq (\underline{J} - \overline{\vartheta})\|s_i\|. \tag{2.4}$$

Similarly, (2.1) implies

$$\|A_i s_i\| \leq (1+\overline{\omega})\|f_i\|. \tag{2.5}$$

Coupling both these inequalities, we obtain

$$\|s_i\| \leq \frac{1+\overline{\omega}}{\underline{J} - \overline{\vartheta}}\|f_i\|, \tag{2.6}$$

so that we can write

$$\begin{aligned} \|J_i s_i + f_i\| &\leq \|A_i s_i + f_i\| + \|(J_i - A_i)s_i\| \leq \overline{\omega}\|f_i\| + \overline{\vartheta}\|s_i\| \\ &\leq \left(\overline{\omega} + \overline{\vartheta}\frac{1+\overline{\omega}}{\underline{J} - \overline{\vartheta}}\right)\|f_i\| \triangleq \overline{\eta}\|f_i\|. \end{aligned}$$

Using the inequality $0 \leq \overline{\vartheta} < (1/2)(1-\overline{\omega})\underline{J}$, we get $0 \leq \overline{\eta} < 1$. □

Lemma 1 shows that the inexactness of the Jacobian matrix can be transformed to the inexactness of the solution to the linear system, so that almost all theoretical results concerning the inexact Newton method (e.g. results from [7] or [18]) can be used when assumption A4 is satisfied. Unfortunately, assumption A4 can be neither verified, if the Jacobian matrix is not known, nor guaranteed in the general case. Therefore, we have to use a different approach for building a globally convergent algorithm. One such possibility is the application of a suitable restarting strategy. If we apply the simple decision

D3: If $A_i \neq J_i$ *and (2.2) has been violated in* $\overline{j}_1$ *consecutive Armijo steps, then set* $A_i = J_i$ *and repeat the iteration,*

we obtain either $A_i = J_i$, so that a theory developed for the inexact Newton method can be used, or $A_i \neq J_i$ and $\alpha_i \geq \underline{\beta}^{\overline{j}_1}$, which eliminate assumption A4 from the proof of global convergence (cf. Theorem 1). The following algorithm realizes above ideas:

Algorithm 1

Data: $0 < \underline{\beta} \leq \overline{\beta} < 1$, $0 < \underline{\rho} < 1$, $0 \leq \overline{\omega} < 1$, $\overline{\varepsilon} > 0$, $\overline{i} > 0$, $0 < \underline{j} \leq \overline{j}_1 \leq \overline{j}_2$, $0 < \overline{k} \leq \infty$ (∞ is allowed).

Step 1: Initiation. Choose an initial point $x_1 \in \mathcal{R}^n$ and compute the vector $f_1 := f(x_1)$. Set $k := 1$ and $i := 1$.

Step 2: Test on convergence. If $\|f_i\| \leq \overline{\varepsilon}$ then terminate the computations (the solution is obtained). If $i > \overline{i}$ then terminate the computations (too many iterations).

Step 3: Direction determination. If $k = 1$ then compute the matrix $J_i := J(x_i)$ and set $A_i = J_i$. Determine $0 \leq \overline{\omega}_i \leq \overline{\omega}$ and compute the vector $s_i \in \mathcal{R}^n$ satisfying the condition $\|A_i s_i + f_i\| \leq \overline{\omega}_i \|f_i\|$, e.g. by Algorithm 2 described below.

Step 4: Backtracking. (a) Set $\alpha_i^1 := 1$ and $j := 1$.

(b) Set $x_{i+1} := x_i + \alpha_i^j s_i$ and compute $f_{i+1} := f(x_{i+1})$. If (2.2) holds then go to Step 5.

(c) If $k = 1$ and $j > \overline{j}_2$, then terminate the computations (the algorithm fails). If $k > 1$ and $j > \overline{j}_1$, then set $k := 1$ and go to Step 3. Otherwise select the value $\underline{\beta}\alpha_i^j \leq \alpha_i^{j+1} \leq \overline{\beta}\alpha_i^j$, set $j := j + 1$ and go to Step 4b.

Step 5: Update. If $j \leq \underline{j}$ and $k \leq \overline{k}$ then compute the matrix A_{i+1} using some Newton-like method, set $k := k + 1$, $i := i + 1$ and go to Step 2. If $j > \underline{j}$ or $k > \overline{k}$ then set $k := 1$, $i := i + 1$ and go to Step 2.

Now we prove that the sequence of points $x_i \in \mathcal{R}^n$, $i \in \mathcal{N}$, generated by Algorithm 1, is globally convergent provided the failure in Step 4 was not indicated. Moreover, we formulate conditions for eliminating this failure. Since required results cannot be easily found in [7] or [18] we give relatively short complete proofs.

Theorem 1 *Let the Jacobian matrix of the function $f : \mathcal{R}^n \to \mathcal{R}^n$ be defined on $\mathcal{D}(\overline{F}, \overline{\Delta})$ and let $x_i \in \mathcal{R}^n$, $i \in \mathcal{N}$, be a sequence generated by Algorithm 1, which does not fail in Step 4. Then $f_i \to 0$. If, in addition, A5 holds, then $x_i \to x^\star$ and $f(x^\star) = 0$.*

Proof Using (2.2) we get

$$\begin{aligned} \|f_i\|(\|f_{i+1}\| - \|f_i\|) &\leq \frac{1}{2}(\|f_{i+1}\| + \|f_i\|)(\|f_{i+1}\| - \|f_i\|) = F_{i+1} - F_i \\ &\leq -2\underline{\rho}(1 - \overline{\omega})\alpha_i F_i \leq -\underline{\rho}(1 - \overline{\omega})\underline{\beta}^{\overline{j}_2}\|f_i\|^2. \end{aligned}$$

Thus

$$\|f_{i+1}\| \leq (1 - \underline{\rho}(1 - \overline{\omega})\underline{\beta}^{\overline{j}_2})\|f_i\| \stackrel{\Delta}{=} \overline{\lambda}\|f_i\|,$$

where $0 < \overline{\lambda} < 1$, and, therefore

$$\sum_{i=1}^{\infty} \|f_i\| = \frac{1}{1-\overline{\lambda}} \|f_1\| < \infty,$$

which implies $f_i \to 0$. If A5 holds, then $\|A_i s_i\| \geq \underline{A}\|s_i\|$, which together with (2.5) gives

$$\sum_{i=1}^{\infty} \|x_{i+1} - x_i\| = \sum_{i=1}^{\infty} \alpha_i \|s_i\| \leq \frac{1+\overline{\omega}}{\underline{A}} \sum_{i=1}^{\infty} \|f_i\| < \infty$$

so that the sequence x_i, $i \in \mathcal{N}$, satisfies the Cauchy condition. Therefore, $x_i \to x^\star$, which together with $f_i \to 0$ gives $f(x^\star) = 0$. □

Theorem 2 *Let assumptions A1 - A3 be satisfied, let $x_i \in \mathcal{L}(\overline{F})$, $A_i = J(x_i)$ and let D1 hold. Then integer $\overline{j}_2$, independent on $i \in \mathcal{N}$, exists, such that the Armijo rule D2 finds, after at most $\overline{j}_2$ steps, a steplength $\alpha_i \geq \underline{\beta}^{\overline{j}_2}$ satisfying (2.2) with a given $0 < \rho < 1$.*

Proof (a) Assume that (2.2) does not hold with $x_{i+1} = x_i + \alpha_i s_i$, i.e.

$$F(x_i + \alpha_i s_i) - F(x_i) > -2\underline{\rho}(1-\overline{\omega})\alpha_i F_i = -\underline{\rho}(1-\overline{\omega})\alpha_i \|f_i\|^2.$$

On the other hand, using assumptions A1, A2 and the inequality

$$g_i^T s_i = f_i^T (J_i s_i + f_i) - f_i^T f_i \leq -(1-\overline{\omega})\|f_i\|^2, \tag{2.7}$$

which holds when $A_i = J_i$, we can write

$$\begin{aligned} F(x_i + \alpha_i s_i) - F(x_i) &= \alpha_i s_i^T g(x_i + \mu\alpha_i s_i) \\ &\leq \alpha_i \left(g_i^T s_i + \|s_i\| \|g(x_i + \mu\alpha_i s_i) - g(x_i)\| \right) \\ &\leq \alpha_i \left(-(1-\overline{\omega})\|f_i\|^2 + \alpha_i(\overline{J}^2 + \overline{LF})\|s_i\|^2 \right), \end{aligned}$$

where $0 \leq \mu \leq 1$, since

$$\begin{aligned} \|g(x_i + \mu\alpha_i s_i) - g(x_i)\| &= \|J^T(x_i + \mu\alpha_i s_i) f(x_i + \mu\alpha_i s_i) - J^T(x_i) f(x_i)\| \\ &\leq \|J^T(x_i + \mu\alpha_i s_i)(f(x_i + \mu\alpha_i s_i) - f(x_i))\| \\ &\quad + \|(J^T(x_i + \mu\alpha_i s_i) - J^T(x_i)) f(x_i)\| \\ &\leq \overline{J}\|(f(x_i + \mu\alpha_i s_i) - f(x_i)\| + \overline{L}\mu\alpha_i \|s_i\| \|f_i\| \\ &= \overline{J}\| \int_0^1 J(x_i + \tau\mu\alpha_i s_i)\mu\alpha_i s_i d\tau\| + \overline{L}\mu\alpha_i \|s_i\| \|f_i\| \\ &\leq (\overline{J}^2 + \overline{LF})\alpha_i \|s_i\|. \end{aligned}$$

By coupling both of these inequalities and using relation (2.6), where we set $\overline{\vartheta} = 0$, we obtain

$$(1 - \overline{\omega} - \underline{\rho}(1-\overline{\omega}))\|f_i\|^2 < \alpha_i(\overline{J}^2 + \overline{LF})\|s_i\|^2 \leq \alpha_i(\overline{J}^2 + \overline{LF}) \left(\frac{1+\overline{\omega}}{\underline{J}} \right)^2 \|f_i\|^2,$$

so that $\alpha_i > \underline{\alpha}$, where

$$0 < \underline{\alpha} = \frac{(1-\underline{\rho})(1-\overline{\omega})\underline{J}^2}{(\overline{J}^2 + \overline{LF})(1+\overline{\omega})^2} < 1. \tag{2.8}$$

(b) Let $\overline{j}_2$ be the lowest integer so that $\overline{\beta}^{\overline{j}_2} \leq \underline{\alpha}$. Since $\underline{\beta}^{\overline{j}_2} \leq \alpha_i \leq \overline{\beta}^{\overline{j}_2}$ holds after $\overline{j}_2$ Armijo steps, then (2.2) necessarily holds after at most $\overline{j}_2$ Armijo steps by part (a) of the proof. □

Theorem 2 can be immediately applied to Algorithm 1. Since the failure in Step 4 can be indicated only if $A_i = J_i$, we can eliminate this case by choosing the sufficiently large integer $\overline{j}_2$, namely $\overline{\beta}^{\overline{j}_2} \leq \underline{\alpha}$, where $\underline{\alpha}$ is given by (2.8). Estimation (2.8) is usually unnecessarily strong and Algorithm 1 works well in practice with a relatively small value $\overline{j}_2 = 10$ as it is demonstrated in Section 4.

Now we focus our attention on details which are necessary for the implementation of descent methods. First we state several comments concerning Algorithm 1:

1) Matrices J_i, $i \in \mathcal{N}$, occuring in Step 3, can be computed either analyticaly or by automatic differentiation or by numerical differentiation. We used the last possibility in our computational experiments to make the Newton method comparable with other Newton-like methods (numerical differentiation described in Section 3 is very efficient for large sparse systems). Notice that the matrices J_i, $i \in \mathcal{N}$, may not be computed explicitly if a transpose-free iterative method is used for the direction determination. In this case, we obtain a matrix-free method that uses numerical differentiation instead of multiplication by the Jacobian matrix.

2) The inequality $0 \leq \overline{\omega}_i \leq \overline{\omega} < 1$, required in Step 3, can be easily satisfied by setting $\overline{\omega}_i = \overline{\omega}$. Nevertheless, a more careful choice of $\overline{\omega}_i$ can slightly improve the efficiency of the inexact Newton-like method. We have used the value $\overline{\omega}_i = \min(\max(\|f_i\|^\nu, \gamma(\|f_i\|/\|f_{i-1}\|)^\alpha), 1/i, \overline{\omega})$, with $\nu = 1/2$, $\gamma = 1$, $\alpha = (1+\sqrt{5})/2$, in our numerical experiments. This choice is a combination of values introduced in [16] and [19] and it implies superlinear convergence of the method, since $\overline{\omega}_i \to 0$ as $i \to \infty$ (see [15]).

3) Experimentally, we have found the values $\underline{j} = 1$, $\overline{j}_1 = 5$, $\overline{j}_2 = 10$ suitable ones in Step 4c. The value $\underline{j}$ has no theoretical importance, it controls a frequency of restart in case the Newton-like method might be inefficient. An experience shows that greater values of $\underline{j}$ increase the total computational time.

4) The value $\underline{\beta}\alpha_i^j \leq \alpha_i^{j+1} \leq \overline{\beta}\alpha_i^j$, computed in Step 4c, can be determined by constant reduction or by more sophisticated procedures such as quadratic or cubic interpolation. We examined all these possibilities and found constant reduction with $\underline{\beta} = \overline{\beta} = 1/2$ a suitable robust strategy for our collection of test problems.

5) If $\overline{k} = \infty$, then restarting is triggered only by backtracking failures. The finite value $\overline{k}$ is essential for limited memory quasi-Newton methods which cannot store more than $O(\overline{k})$ vectors. Setting $\overline{k} = 0$, we obtain the discrete Newton method.

Now we concentrate our attention on the determination of the direction vector. The vector $s_i \in \mathcal{R}^n$, $i \in \mathcal{N}$, satisfying the inequality $\|A_i s_i + f_i\| \leq \overline{\omega}\|f_i\|$ is most frequently

obtained as an approximate solution to the linear subproblem $A_i s + f_i = 0$ using some iterative method. In order to simplify the notation we omit the outer iteration index i in the remainder of this section, so that we write A, f, x instead of A_i, f_i, x_i. On the other hand, we use the inner iteration index j for the description of iterative methods for linear subproblems. To satisfy the condition $\|As + f\| \leq \overline{\omega}\|f\|$, for an arbitrary $0 \leq \overline{\omega} < 1$, we need iterative methods which terminate after a finite number of steps. Moreover, computational experiments show that it is advantageous when these methods generate a sequence of iterates s_j, $j \in \mathcal{N}$, and corresponding residual vectors $r_j = As_j + f$, $j \in \mathcal{N}$, so that the norms $\|r_j\|$, $j \in \mathcal{N}$, do not increase. This requirement can be fulfilled by the choice of some residual minimizing or smoothed conjugate gradient-type method. Moreover, since the system matrix A is not always explicitly known but can be given by the difference formula, we consider only the iterative methods which do not involve multiplication by the transpose of the matrix A (transpose-free methods).

One of the best-known and most widely used schemes of this type is the GMRES method presented by Saad and Schultz in [34]. Unfortunately, this method uses long recurrences ($O(n^3)$ operations and $O(n^2)$ storage in the unrestarted case or $O(m^2 n)$ operations and $O(mn)$ storage in the m-steps restarted case) so that it may not be efficient for large-scale problems. We have had a good experience with the preconditioned smoothed CGS method, presented in [39] and given by the following algorithm.

Algorithm 2. Preconditioned smoothed CGS method.

Compute $s = -C^{-1}f$ *and* $r = As + f$. *If* $\|r\| \leq \overline{\omega}\|f\|$, *then stop.*
Otherwise set $s_1 = 0$, $\overline{s}_1 = 0$, $r_1 = f$, $\overline{r}_1 = f$, $p_1 = f$, $u_1 = f$.
for $j = 1, 2, 3, \ldots$ **do**
If $\|r_j\| \leq \overline{\omega}\|f\|$, *then set* $s = s_j$, $r = r_j$ *and stop. Otherwise set*
$v_j = AC^{-1}p_j$, $\alpha_j = f^T\overline{r}_j / f^T v_j$,
$q_j = u_j - \alpha_j v_j$,
$\overline{s}_{j+1} = \overline{s}_j + \alpha_j C^{-1}(u_j + q_j)$,
$\overline{r}_{j+1} = \overline{r}_j + \alpha_j AC^{-1}(u_j + q_j)$, $\beta_j = f^T\overline{r}_{j+1} / f^T\overline{r}_j$,
$u_{j+1} = \overline{r}_{j+1} + \beta_j q_j$,
$p_{j+1} = u_{j+1} + \beta_j(q_j + \beta_j p_j)$,
$[\lambda_j, \mu_j]^T = \arg \min_{[\lambda,\mu]^T \in \mathcal{R}^2} \|\overline{r}_{j+1} + \lambda(r_j - \overline{r}_{j+1}) + \mu v_j\|$,
$s_{j+1} = \overline{s}_{j+1} + \lambda_j(s_j - \overline{s}_{j+1}) + \mu_j C^{-1}p_j$,
$r_{j+1} = \overline{r}_{j+1} + \lambda_j(r_j - \overline{r}_{j+1}) + \mu_j v_j$.
end do

The matrix C serves for preconditioning. We used an incomplete LU decomposition of the matrix $A + \varepsilon\mathrm{diag}(A)$ as a preconditioner. Here $\mathrm{diag}(A)$ is a diagonal matrix which has the same diagonal as the matrix A and $\varepsilon > 0$ is a small number. Since the two parameter-minimal residual smoothing of the original CGS method [36] is used, the sequence of residual norms is non-increasing. The smoothed CGS method uses

short recurrences ($O(n)$ operations and storage requirement per iteration step), but it can break down if either $f^T\overline{r}_j = 0$ or $f^T v_j = 0$. The solution of a linear system is obtained after at most n iterations (if breakdown does not occur and if rounding errors do not deteriorate the finite termination of the method). Note, that breakdown rarely appears. We have not met this situation in any of our computational experiments.

3 Review of Newton-like methods for nonlinear equations

In this section, we describe a set of methods for solving systems of nonlinear equations, which can be realized by Algorithm 1. All methods differ from each other only by the approximation of the Jacobian matrix $J(x)$. Since we need the true Jacobian matrix for restarts, we begin with the numerical differentiation. For convenience, we denote the sparsity pattern of $J(x)$ by $\mathcal{S}$. Then $(i,j) \in \mathcal{S}$ if, and only if, $J_{ij}(x) \neq 0$ (structurally).

The Jacobian matrix can be determined numerically by using two different ways. The first way, elementwise differentiation, is based on the approximation

$$J_{ij}(x) = \frac{f_i(x + \delta_j e_j) - f_i(x)}{\delta_j}, \tag{3.1}$$

for all $(i,j) \in \mathcal{S}$. Thus we need m scalar function evaluations (i.e. m/n equivalent vector function evaluations) where m is the number of nonzero elements in $J(x)$.

The second way, groupwise differentiation, is based on a division of columns of $J(x)$ into groups $\mathcal{C}_k$, $1 \leq k \leq p$, so that each column belongs to only one group and, moreover, $(i,j_1) \in \mathcal{S}$, $(i,j_2) \in \mathcal{S}$, $j_1 \neq j_2$ imply $j_1 \in \mathcal{C}_{k_1}$, $j_2 \in \mathcal{C}_{k_2}$, $k_1 \neq k_2$. Then, for each group $\mathcal{C}_k$, $1 \leq k \leq p$, we compute the difference $f(x + \sum_{j \in \mathcal{C}_k} \delta_j e_j) - f(x)$ and set

$$J_{ij}(x) = \frac{e_i^T (f(x + \sum_{j \in \mathcal{C}_k} \delta_j e_j) - f(x))}{\delta_j}, \tag{3.2}$$

for all $(i,j) \in \mathcal{S} \cap \mathcal{C}_k$ (we use the notation $\mathcal{S} \cap \mathcal{C}_k = \{(i,j) \in \mathcal{S} : j \in \mathcal{C}_k\}$ and $\mathcal{S} \backslash \mathcal{C}_k = \{(i,j) \in \mathcal{S} : j \notin \mathcal{C}_k\}$). Therefore, we need p vector function evaluations. Since the number of groups cannot be less then the number of nonzero elements in an arbitrary row, the number of vector function evaluations is usually slightly greater than the one connected with the elementwise differentiation. On the other hand, the computation can now be organized better (the expressions which are common for all scalar functions can be computed only p times), so that the groupwise differentiation is usually faster. Groupwise differentiation was first proposed in [13]. The optimum division of columns into groups and the equivalent graph coloring problem were studied in [11]. Efficient implementation of the resulting algorithm is given in [12]. We used this algorithm in all of our experiments (only the discrete Newton method was tested with both the elementwise and the groupwise differentiation).

Now we are in the position to describe individual methods for solving systems of nonlinear equations. The notation refers to Algorithm 1.

1) The discrete Newton method with elementwise differentiation (DNE). This method uses the value $\overline{k} = 0$. Elements of $J(x)$ are computed by (3.1).

2) The discrete Newton method with groupwise differentiation (DNG). This method uses the value $\overline{k} = 0$. Elements of $J(x)$ are computed by (3.2).

3) The Broyden-Schubert (BS) method. This method was introduced in [9] and [35]. If $k < \overline{k}$, then we use the update

$$A_{ij}^+ = A_{ij} + \frac{e_i^T(y - Ad)d_j}{\sum_{(i,k)\in\mathcal{S}} d_k^2}$$

$\forall (i,j) \in \mathcal{S}$. Here $d = x^+ - x$ and $y = f(x^+) - f(x)$. Clearly, $A_{ij}^+ = 0$ if $(i,j) \notin \mathcal{S}$. We set $\overline{k} = \infty$ in our computational experiments.

4) The Bogle-Perkins (BP) method. This method was proposed in [4], If $k < \overline{k}$, then we use the update

$$A_{ij}^+ = A_{ij} + \frac{e_i^T(y - Ad)A_{ij}^2 d_j}{\sum_{(i,k)\in\mathcal{S}} A_{ik}^2 d_k^2}$$

$\forall (i,j) \in \mathcal{S}$. Here $d = x^+ - x$ and $y = f(x^+) - f(x)$. Clearly, $A_{ij}^+ = 0$ if $(i,j) \notin \mathcal{S}$. We set $\overline{k} = \infty$ in our computational experiments.

5) The Li (LI) method. This method, proposed in [25], is based on the groupwise differentiation. If $k < \overline{k}$, then only one group of columns is updated by numerical differentiation. Other columns remain unchanged. In other words, we set $l := l + 1$ if $l < p$ and $l := 1$ if $l = p$ ($l = 0$ is the starting value) and then substitute

$$A_{ij}^+ = \frac{e_i^T(f(x + \sum_{j\in\mathcal{C}_l} \delta_j e_j) - f(x))}{\delta_j},$$

$\forall (i,j) \in \mathcal{S} \cap \mathcal{C}_l$ and

$$A_{ij}^+ = A_{ij},$$

$\forall (i,j) \in \mathcal{S}\backslash\mathcal{C}_l$. Clearly, $A_{ij}^+ = 0$ if $(i,j) \notin \mathcal{S}$. We set $\overline{k} = \infty$ in our computational experiments.

6) The combination (LIBS) of the Li and the Broyden-Schubert methods. This method is again based on the groupwise differentiation. If $k < \overline{k}$, then only one group of columns is updated by numerical differentiation. Other columns are updated using the Broyden-Schubert algorithm. In other words, we set $l := l + 1$ if $l < p$ and $l := 1$ if $l = p$ ($l = 0$ is the starting value) and then substitute

$$A_{ij}^+ = \frac{e_i^T(f(x + \sum_{j\in\mathcal{C}_l} \delta_j e_j) - f(x))}{\delta_j},$$

$\forall (i,j) \in \mathcal{S} \cap \mathcal{C}_l$ and

$$A^+_{ij} = A_{ij} + \frac{e_i^T(y - Ad)d_j}{\sum_{(i,k)\in\mathcal{S}\setminus\mathcal{C}_l} d_k^2},$$

$\forall (i,j) \in \mathcal{S}\setminus\mathcal{C}_l$. Here $d = x^+ - x$ and $y = f(x^+) - f(x)$. Clearly, $A^+_{ij} = 0$ if $(i,j) \notin \mathcal{S}$. We set $\overline{k} = \infty$ in our computational experiments.

7) The modified Newton (MN) method. We set $A^+ := A$, if $k < \overline{k}$. This method needs a finite value $\overline{k}$. The value $\overline{k} = 5$ was obtained experimentally.

8) The row scaling (RS) method. We set $A^+ := DA$, if $k < \overline{k}$, where the diagonal matrix D is determined from the quasi-Newton condition $DAd = y$, i.e.

$$e_i^T D e_i = \frac{e_i^T y}{e_i^T A d}$$

for all $1 \le i \le n$ (here $d = x^+ - x$ and $y = f(x^+) - f(x)$). The row scaling method was proposed in [20] in connection with the complete LU decomposition. This method needs a finite value $\overline{k}$. The value $\overline{k} = 5$ was obtained experimentally.

9) The limited memory good Broyden (LMB) method. This method is a modification of the good Broyden method introduced in [8] and it is based on a compact representation of quasi-Newton matrices proposed in [10]. Denote by $D = [d, d_{-1}, \ldots, d_{-k}]$ and $Y = [y, y_{-1}, \ldots, y_{-k}]$ the matrices constructed from the last k differences $d = x^+ - x$, $d_{-1} = x - x_{-1}$, ..., $d_{-k} = x_{1-k} - x_{-k}$ and $y = f(x^+) - f(x)$, $y_{-1} = f(x) - f(x_{-1})$, ..., $y_{-k} = f(x_{1-k}) - f(x_{-k})$ respectively, and define the upper triangular matrix

$$R = \begin{bmatrix} d^T d, & d^T d_{-1}, & \ldots, & d^T d_{-k} \\ 0, & d_{-1}^T d_{-1} & \ldots, & d_{-1}^T d_{-k} \\ \ldots, & \ldots, & \ldots, & \ldots \\ 0, & 0, & \ldots, & d_{-k}^T d_{-k} \end{bmatrix}.$$

Then, if $k < \overline{k}$, we set

$$A^+ = A_{-k} + (Y - A_{-k}D)R^{-1}D^T.$$

The limited memory Broyden method needs a finite value $\overline{k}$. We obtained $\overline{k} = 5$ experimentally.

10) The limited memory column update (LMC) method. This method is a modification of the column update method introduced in [29] and is based on a compact representation of quasi-Newton matrices proposed in [10]. Let D and Y be the same matrices as in the previous case. Denote $e = \arg\max_{e_i} |e_i^T d|$, $e_{-1} = \arg\max_{e_i} |e_i^T d_{-1}|$, ..., $e_{-k} = \arg\max_{e_i} |e_i^T d_{-k}|$ ($\arg\max$ is taken over all e_i, $1 \le i \le n$) set $E = [e, e_{-1}, \ldots, e_{-k}]$ and define the upper triangular matrix

$$R = \begin{bmatrix} e^T d, & e^T d_{-1}, & \ldots, & e^T d_{-k} \\ 0, & e_{-1}^T d_{-1} & \ldots, & e_{-1}^T d_{-k} \\ \ldots, & \ldots, & \ldots, & \ldots \\ 0, & 0, & \ldots, & e_{-k}^T d_{-k} \end{bmatrix}.$$

Then, if $k < \overline{k}$, we set

$$A^+ = A_{-k} + (Y - A_{-k}D)R^{-1}E^T$$

Note that the vectors e, e_{-1}, ..., e_{-k} do not need to be stored. We only use indices of their unique nonzero elements. The limited memory column update method needs a finite value $\overline{k}$. We obtained $\overline{k} = 5$ experimentally.

11) The limited memory inverse column update (LMI) method. This method, which was introduced in [30], uses an approximation $S = A^{-1}$ of the inverse Jacobian matrix $J^{-1}(x)$. Therefore, if $k < \overline{k}$, we simply set $s := -Sf$ instead of using Algorithm 2. Denote $e_{-1} = \arg\max_{e_i} |e_i^T y_{-1}|$, ..., $e_{-k} = \arg\max_{e_i} |e_i^T y_{-k}|$ (arg max is taken over all e_i, $1 \le i \le n$). Then the vector Sf can be computed by the formula

$$Sf = S_{-k}f + \frac{e_{-1}^T f}{e_{-1}^T y_{-1}} v_{-1} + \ldots + \frac{e_{-k}^T f}{e_{-k}^T y_{-k}} v_{-k},$$

where $v_{-1} = d_{-1} - S_{-1}y_{-1}$, ..., $v_{-k} = d_{-k} - S_{-k}y_{-k}$. These vectors can be computed recursively by the formula

$$Sy = S_{-k}y + \frac{e_{-1}^T y}{e_{-1}^T y_{-1}} v_{-1} + \ldots + \frac{e_{-k}^T y}{e_{-k}^T y_{-k}} v_{-k}.$$

In both of these formulae we use the matrix $S_{-k} = (L_{-k}U_{-k})^{-1}$, where $L_{-k}U_{-k}$ is the incomplete LU decomposition of the Jacobian matrix $J(x_{-k})$. Note that the vectors e_{-1}, ..., e_{-k} do not need to be stored. We only use indices of their unique nonzero elements. The limited memory column update method needs a finite value $\overline{k}$. We obtained $\overline{k} = 6$ experimentally.

12) The discrete Newton method with successive differentiation (DNS). This method, proposed in [26], does not use Jacobian matrices. The products $Av = Jv$, which appear in Algorithm 2, are replaced by the numerical differentiation

$$Av = \frac{f(x + v\delta/\|v\|) - f(x)}{\delta/\|v\|}$$

where δ is a small difference (usually $\delta = 10^{-8}$ for a double precision arithmetic). Since the Jacobian matrix is not computed explicitly, we cannot use the incomplete LU decomposition of the Jacobian matrix as a preconditioner. Instead, we numerically compute, using differences, the tridiagonal part of the Jacobian matrix and then apply this tridiagonal matrix as a preconditioner.

4 Numerical experiments

In this section we present results of a comparative study of the Newton-like methods, described in Section 3, which were realized as Armijo-type descent methods (Algorithm 1) with inexact iterative solution of linear subproblems by the preconditioned

smoothed CGS method (Algorithm 2). These methods were implemented by using the modular interactive system for universal functional optimization UFO [27]. We used the values $\underline{\beta} = \overline{\beta} = 0.5$, $\underline{\rho} = 10^{-4}$, $\overline{\omega} = 0.4$, $\overline{\varepsilon} = 10^{-16}$, $\overline{i} = 200$, $\underline{j} = 1$, $\overline{j}_1 = 5$, $\overline{j}_2 = 10$ in Algorithm 1. All test results were obtained by using 30 sparse problems. Names and sizes of these problems, together with their sources, are given in Table 1 (n is the number of equations and m is the number of nonzeroes in the Jacobian matrix).

Table 1: *Test problems for nonlinear equations.*

No.	Problem	n	m
1	Countercurrent Reactor Problem 1, [4]	5000	19996
2	Countercurrent Reactor Problem 2, [4]	5000	24993
3	Trigonometric System, [38]	5000	25000
4	Trigonometric-Exponential System - Trigexp 1, [38]	5000	14998
5	Trigonometric-Exponential System - Trigexp 2, [38]	4999	19993
6	Singular Broyden System, [20]	5000	14998
7	Tridiagonal System, [25]	5000	14998
8	Five-Diagonal System, [25]	5000	24994
9	Seven-Diagonal System, [25]	5000	34988
10	Structured Jacobian Problem, [20]	5000	39984
11	Extended Freudenstein and Roth Problem, [3]	5000	10000
12	Extended Powell Singular Problem, [31]	5000	10000
13	Extended Cragg and Levy Problem, [31]	5000	8750
14	Broyden Tridiagonal System, [31]	5000	14998
15	Broyden Banded System, [31]	5000	34984
16	Extended Powell Badly Scaled Problem, [31]	5000	10000
17	Extended Wood Function, [22]	5000	12500
18	Tridiagonal System, [3]	5000	14998
19	Discrete Boundary Value Problem, [31]	5000	14998
20	Discrete Boundary Value Problem, [2]	5000	14998
21	Troesch Problem, [33]	5000	14998
22	Flow in a Channel, [1]	5000	24994
23	Swirling Flow, [1]	5000	34998
24	Bratu problem, [23]	4900	24220
25	Poisson Problem, [20]	4900	24220
26	Poisson Problem, [28]	4900	24220
27	Porous Medium Problem, [19]	4900	24220
28	Convection-Difussion Problem, [24]	4900	24220
29	Nonlinear Biharmonic Problem, [21]	2500	31504
30	Driven Cavity Problem, [23]	2500	31504

The first 21 problems have a standard form and their complete description can be found in the cited references while the last 8 problems require more detailed comments:
Problem 22: This is a finite difference analogue of the following nonlinear ordinary differential equation

$$u'''' = R(u' u'' - u u'''), \quad R = 500$$

over the unit interval Ω with the boundary conditions $u(0) = 0$, $u'(0) = 0$, $u(1) = 1$, $u'(1) = 0$. We used standard 5-point finite differences on an uniform grid having 5000 internal nodes. The initial approximate solution was a discretization of $u_0(x) = (x - 1/2)^2$.

Problem 23: This is a finite difference analogue of the following system of two nonlinear ordinary differential equations

$$u^{''''} + R(uu^{'''} + vv^{'}) = 0$$
$$v^{''} + R(uv^{'} + u^{'}v) = 0, \quad R = 500$$

over the unit interval Ω with the boundary conditions $u(0) = u^{'}(0) = u(1) = u^{'}(1) = 0$, $v(0) = -1$, $v(1) = 1$. We used standard 5-point finite differences on an uniform grid having 2500 internal nodes. The initial approximate solution was a discretization of $u_0(x) = (x - 1/2)^2$ and $v_0(x) = x - 1/2$.

Problem 24: This is a finite difference analogue of the following nonlinear partial differential equation

$$\Delta u + R\exp(u) = 0, \quad R = 6.8$$

over the unit square Ω with Dirichlet boundary conditions $u = 0$ on $\partial\Omega$. We used standard 5-point finite differences on a uniform grid having 70×70 internal nodes. The initial approximate solution was a discretization of $u_0(x, y) = 0$.

Problem 25: This is a finite difference analogue of the following nonlinear partial differential equation

$$\Delta u = \frac{u^3}{1 + x^2 + y^2}$$

over the unit square Ω with Dirichlet boundary conditions $u(0, y) = 1$, $u(1, y) = 2 - \exp(y)$, $u(x, 0) = 1$, $u(x, 1) = 2 - \exp(x)$. We used standard 5-point finite differences on a uniform grid having 70×70 internal nodes. The initial approximate solution was a discretization of $u_0(x, y) = -1$.

Problem 26: This is a finite difference analogue of the following nonlinear partial differential equation

$$\Delta u + \sin(2\pi u) + \sin\left(2\pi\frac{\partial u}{\partial x}\right) + \sin\left(2\pi\frac{\partial u}{\partial y}\right) + f(x, y) = 0,$$

where $f(x, y) = 1000((x - 1/4)^2 + (y - 3/4)^2)$, over the unit square Ω with Dirichlet boundary conditions $u = 0$ on $\partial\Omega$. We used standard 5-point finite differences on a uniform grid having 70×70 internal nodes. The initial approximate solution was a discretization of $u_0(x, y) = 0$.

Problem 27: This is a finite difference analogue of the following nonlinear partial differential equation

$$\Delta u^2 + R\left(\frac{\partial u^3}{\partial x} + f(x, y),\right) = 0, \quad R = 50$$

where $f(1/71, 1/71) = 1$ and $f(x, y) = 0$ for $(x, y) \neq (1/71, 1/71)$, over the unit square Ω with Dirichlet boundary conditions $u(0, y) = 1$, $u(1, y) = 0$, $u(x, 0) = 1$, $u(x, 1) = 0$. We used standard 5-point finite differences on a uniform grid having 70×70 internal nodes. The initial approximate solution was a discretization of $u_0(x, y) = 1 - xy$.

Problem 28: This is a finite difference analogue of the following nonlinear partial differential equation

$$\Delta u - Ru\left(\frac{\partial u}{\partial x} + \frac{\partial u}{\partial y}\right) + f(x,y) = 0, \quad R = 20,$$

where $f(x,y) = 2000x(1-x)y(1-y)$, over the unit square Ω with Dirichlet boundary conditions $u = 0$ on $\partial\Omega$. We used standard 5-point finite differences on a uniform grid having 70×70 internal nodes. The initial approximate solution was a discretization of $u_0(x,y) = 0$.

Problem 29: This is a finite difference analogue of the following nonlinear partial differential equation

$$\Delta\Delta u + R\left(\max(0,u) + \mathrm{sign}(x - \frac{1}{2})\right) = 0, \quad R = 500$$

over the unit square Ω with the boundary conditions $u = 0$ on $\partial\Omega$ and $\partial u(0,y)/\partial x = 0$, $\partial u(1,y)/\partial x = 0$, $\partial u(x,0)/\partial y = 0$, $\partial u(x,1)/\partial y = 0$. We used standard 13-point finite differences on a shifted uniform grid having 50×50 internal nodes [23]. The initial approximate solution was a discretization of $u_0(x,y) = 0$.

Problem 30: This is a finite difference analogue of the following nonlinear partial differential equation

$$\Delta\Delta u + R\left(\frac{\partial u}{\partial y}\frac{\partial \Delta u}{\partial x} - \frac{\partial u}{\partial x}\frac{\partial \Delta u}{\partial y}\right) = 0, \quad R = 500$$

over the unit square Ω with the boundary conditions $u = 0$ on $\partial\Omega$ and $\partial u(0,y)/\partial x = 0$, $\partial u(1,y)/\partial x = 0$, $\partial u(x,0)/\partial y = 0$, $\partial u(x,1)/\partial y = 1$. We used standard 13-point finite differences on a shifted uniform grid having 50×50 internal nodes [23]. The initial approximate solution was a discretization of $u_0(x,y) = 0$.

A summary of results for all of these problems is given in two tables presented below. These tables consist of three parts. The first part introduces, for each Newton-like method, geometric means of individual numbers of iterations IT, function evaluations FV, CGS iterations CG (or complete LU decompositions DC), backtracking steps LS and computational times in seconds. The geometric mean of n_i, $1 \le i \le 30$ was computed by the formula $\left(\prod_{i=1}^{30}(n_i+1)\right)^{1/30} - 1$. The second part introduces total numbers of iterations IT, function evaluations FV, CGS iterations CG (or complete LU decompositions DC), computational times in seconds and possible failures. Failures caused by exceeding upper limits $\overline{i}$ and $\overline{j}_2$ in Algorithm 1 sometimes appeared when problems 1, 2, 5, 8, 17 and 23 were solved. Data for cases of failures are included in the overall statistics. The third part contains total storage requirements for all problems in kilobytes kB. Table 2a corresponds to preconditioned smoothed CGS algorithm. We used an incomplete LU decomposition of the matrix $A + \varepsilon\mathrm{diag}(A)$ as a preconditioner, where $\varepsilon = 0$ for problems 1-28 and $\varepsilon = 10^{-2}$ for ill-conditioned problems 29-30. Table 2b is comparative and it contains results obtained after replacing the preconditioned smoothed CGS algorithm by complete LU decomposition (we used the unsymmetric-pattern multifrontal scheme implemented in the UMFPACK package [14]).

Table 2a: *Results for various Newton-like methods with preconditioned smoothed CGS iterations.*

Method	IT	FV	CG	LS	time	IT	FV	CG	time	fail	kB
DNE	11	59	3	0	10.54	414	2289	1271	524.82	0	1440
DNG	11	63	3	0	9.78	414	2454	1271	507.35	0	1760
BS	17	45	4	5	13.99	657	1997	2456	778.95	2	1760
BP	18	49	5	5	17.94	769	2642	2778	998.54	1	1760
LI	19	85	4	10	16.92	805	5092	3614	1288.88	4	1760
LIBS	16	59	5	5	15.53	757	2556	3504	1179.74	4	1760
MN	21	57	4	4	12.17	776	2332	2287	674.92	2	1760
RS	15	54	4	5	12.63	586	2637	2573	803.07	3	1760
LMB	15	42	20	2	13.80	572	1800	1923	677.89	1	2220
LMC	14	46	18	2	12.74	554	2032	2002	659.82	1	2060
LMI	19	49	3	3	8.77	670	1857	1414	498.62	0	2000
DNS	10	249	32	1	173.99	386	17113	7430	6473.46	9	620

Table 2b: *Results for various Newton-like methods with the complete LU decomposition (UMFPACK).*

Method	IT	FV	DC	LS	time	IT	FV	DC	time	fail	kB
DNE	10	60	10	1	14.75	445	3021	445	790.81	3	6300
DNG	10	64	10	1	14.68	441	3158	441	792.52	3	6460
BS	14	41	14	4	18.11	606	2028	626	1128.94	4	6460
BP	14	41	15	4	20.32	643	2203	664	1065.39	4	6460
LI	17	79	20	9	24.07	765	4998	950	1721.37	6	6460
LIBS	14	55	15	4	19.56	735	2812	788	1269.99	5	6460
MN	20	60	5	4	12.11	795	2850	208	574.52	2	6460
RS	14	47	5	5	10.92	560	2015	206	560.02	2	6500
LMI	14	37	3	2	9.09	537	1383	136	406.07	1	6700

According to the results presented in the above tables, we can make several conclusions (which are, of course, influenced by our collection of test problems):

1) Newton like methods with preconditioned smoothed CGS iterations are competitive (measured by the total computational time) with methods based on complete LU decomposition. On the other hand, the former ones are more robust and have lower storage requirements.

2) Discrete Newton methods (DNE, DNG) are more efficient and more robust than sparse quasi-Newton methods (BS, BP). Methods based on cyclic differentiation (LI, LIBS) were shown as the worst ones (measured by the computational time and the number of failures) among all tested Newton-like methods (with the exception of the derivative free DNS method).

3) Limited memory quasi-Newton methods (LMB, LMC, LMI) are more efficient and more robust than sparse quasi-Newton methods (BS, BP). Particularly, the LMI method was shown as the best one (measured by the computational time and the number of failures) among all tested Newton-like methods.

4) The DNS method is neither efficient nor robust since it cannot be implemented with an efficient preconditioner based on incomplete LU decomposition. Almost all failures were caused by exceeding the maximum number 1200 of function evaluations. However, this method has a minimum storage requirement and sometimes gives very good results.

5) Algorithm 1 is very efficient and robust at least in connection with the LMI method.

Bibliography

[1] Averick, B.M., and Carter, R.G., and Moré, J.J., *The Minpack-2 Test Problem Collection,* Research Report No. ANL/MCS-TM-150, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne 1991.

[2] Alefeld, G., and Gienger, A., and Potra, F., *Efficient Validation of Solutions of Nonlinear Systems,* SIAM Journal on Numerical Analysis, Vol 31, pp. 252-260, 1994.

[3] Bing, Y., and Lin, G., *An Efficient Implementation of Merrill's Method for Sparse of Partially separable systems of Nonlinear Equations,* SIAM Journal on Optimization, Vol 2, pp. 206-221, 1991.

[4] Bogle, I.D.L., and Perkins, J.D., *A New Sparsity Preserving Quasi-Newton Update for Solving Nonlinear Equations,* SIAM Journal on Scientific and Statistical Computations, Vol 11, pp. 621-630, 1990.

[5] Brown, P.N., *A Local Convergence Theory for Combined Inexact-Newton/Finite-Difference Projection Methods,* SIAM Journal on Numerical Analysis, Vol. 24, pp. 407-434, 1987.

[6] Brown, P.N., and Saad, Y., *Hybrid Krylov Methods for Nonlinear Systems of Equations,* SIAM Journal on Scientific and Statistical Computations, Vol. 11, pp. 450-481, 1990.

[7] Brown, P.N., and Saad, Y., *Convergence Theory of Nonlinear Newton-Krylov Algorithms,* SIAM Journal on Optimization, Vol. 4, pp. 297-330, 1994.

[8] Broyden, C.G., *A Class of Methods for Solving Simultaneous Equations,* Mathematics of Computation, Vol. 19, pp. 577-593, 1965.

[9] Broyden, C.G., *The Convergence of an Algorithm for Solving Sparse Nonlinear Systems,* Mathematics of Computation, Vol. 25, pp. 285-294, 1971.

[10] Byrd, R.H, and Nocedal J., and Schnabel R.B., *Representations of Quasi-Newton Matrices and their Use in Limited Memory Methods,* Mathematical Programming, Vol. 63, pp. 129-136, 1994.

[11] Coleman, T.F., and Moré, J.S., *Estimation of Sparse Jacobian and Graph Coloring Problem,* SIAM Journal on Numerical Analysis, Vol. 20, pp. 187-209, 1983.

[12] Coleman, T.F., and Garbow, B.S., and Moré, J.S., *Software for Estimating Sparse Jacobian Matrices,* ACM Transactions of Mathematical Software, Vol. 10, pp. 329-345, 1984.

[13] Curtis, A.R., and Powell, M.J.D., and Reid, J.K., *On the estimation of sparse Jacobian matrices,* IMA Journal of Aplied Mathematics, Vol. 13, pp. 117-119, 1974.

[14] Davis, T.A., *User's Guide for the Unsymmetric-Pattern Multifrontal Package (UMFPACK),* Research Report No. TR-93-020, Computer and Information Sciences Department, University of Florida, Gainesville, Florida, 1993.

[15] Dembo, R.S, Eisenstat, S.C., and Steihaug T., *Inexact Newton Methods,* SIAM J. on Numerical Analysis, Vol. 19, pp. 400-408, 1982.

[16] Dembo, R.S., and Steihaug T., *Truncated Newton Algorithms for Large-Scale Optimization,* Mathematical Programming, Vol. 26, pp. 190-212, 1983.

[17] Dennis, J.E., and Schnabel, R.B., *Numerical Methods for Unconstrained Optimization and Nonlinear Equations,* Prentice-Hall, Englewood Cliffs, New Jersey, 1983.

[18] Eisenstat, S.C., and Walker, H.F., *Globally convergent Inexact Newton Methods,* SIAM Journal on Optimization, Vol. 4, pp. 393-422, 1994.

[19] Eisenstat, S.C., and Walker, H.F., *Choosing the Forcing Terms in an Inexact Newton Method,* SIAM Journal on Scientific Computation, Vol. 17, pp. 16-32, 1996.

[20] Gomez-Ruggiero, M.A., and Martinez, J.M., and Moretti, A.C., *Comparing Algorithms for Solving Sparse Nonlinear Systems of Equations,* SIAM Journal on Scientific and Statistical Computations, Vol. 13, pp. 459-483, 1992.

[21] Hlaváček, L., and Lovíšek, J., *Optimal Design of an Elastic or Elasto-plastic Beam with Unilateral Elastic Foundation and Rigid Supports,* Zeitschrift fur Angevandte Mathematik und Mechanik, Vol. 72, pp. 29-43, 1992.

[22] Incerti, S., and Zirilli, F., and Parisi, V., *Algorithm 111. A Fortran Subroutine for Solving Systems of Nonlinear Simultaneous Equations,* Computer Journal, Vol 24, pp. 87-91, 1981.

[23] Kaporin, I.E., and Axelsson, O., *On a Class of Nonlinear Equation Solvers Based on the Residual Norm Reduction Over a Sequence of Affine Subspaces,* SIAM Journal on Scientific and Statistical Computations, Vol 16, pp. 228-249, 1995.

[24] Kelley, C.T., *Iterative Methods for Linear and Nonlinear Equations,* SIAM, Philadelphia, Pennsylvania, 1995.

[25] Li, G., *Successive Column Correction Algorithms for Solving Sparse Nonlinear Systems of Equations,* Mathematical Programming, Vol. 43, pp. 187-207, 1989.

[26] Lukšan, L., *Inexact Trust Region Method for Large Sparse Systems of Nonlinear Equations,* Journal of Optimization Theory and Applications, Vol. 81, pp. 569-590, 1994.

[27] Lukšan, L., and Šiška, M., and Tůma, M., and Vlček, J., and Ramešová, N., *Interactive System for Universal Functional Optimization (UFO), Version 1994,* Research Report No. V-599, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague, Czech Republic, 1994.

[28] Martinez, J.M., *A Quasi-Newton Method with Modification of One Column per Iteration,* Computing, Vol. 33, pp. 353-362, 1984.

[29] Martinez, J.M., *SOR-Secant Methods,* SIAM Journal on Numerical Analysis, Vol. 31, pp. 217-226, 1994.

[30] Martinez, J.M., and Zambaldi, M.C., *An Inverse Column-Updating Method for Solving Large-Scale Nonlinear Systems of Equations,* Optimization Methods and Software, Vol. 1, pp. 129-140, 1992.

[31] Moré, J.J., and Garbow, B.S., and Hillström, K.E., *Testing Unconstrained Optimization Software,* ACM Transactions on Mathematical Software, Vol. 7, pp. 17-41, 1981.

[32] Powell, M.J.D., *On the Global Convergence of Trust Region Algorithms for Unconstrained Minimization,* Mathematical Programming, Vol. 29, pp. 297-303, 1984.

[33] Roberts, S.M., and Shipman, J.S., *On the Closed Form Solution of Troesch's Problem,* Journal of Computational Physics, Vol. 21, pp. 291-304, 1976.

[34] Saad, Y., Schultz, M., *GMRES a Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems,* SIAM Journal on Scientific and Statistical Computations, Vol. 7, pp. 856-869, 1986.

[35] Schubert, L.K., *Modification of a Quasi-Newton Method for Nonlinear Equations with Sparse Jacobian,* Mathematics of Computation, Vol. 24, pp. 27-30, 1970.

[36] Sonneveld, P., *CGS, a Fast Lanczos-Type Solver for Nonsymmetric Linear Systems,* SIAM Journal on Scientific and Statistical Computations, Vol. 10, pp. 36-52, 1989.

[37] Steihaug, T., *The Conjugate Gradient Method and Trust Regions in Large-Scale Optimization,* SIAM Journal on Numerical Analysis, Vol. 20, pp. 626-637, 1983.

[38] Toint, P.L., *Numerical Solution of Large Sets of Algebraic Equations,* Mathematics of Computation, Vol. 46, pp. 175-189, 1986.

[39] Tong, C.H., *A Comparative Study of Preconditioned Lanczos Methods for Nonsymmetric Linear Systems,* Sandia National Laboratories, Sandia Report No. SAND91-8240B, Livermore, 1992.