Optimization Methods and Software

ISSN: 1055-6788 (Print) 1029-4937 (Online) Journal homepage: http://www.tandfonline.com/loi/goms20

# A nonmonotone hybrid method for nonlinear systems

Maria Grazia Gasparo

*Optimization Meth. & Soft.*, 2000, Vol. 13, pp. 79–94
Reprints available directly from the publisher
Photocopying permitted by license only

# A NONMONOTONE HYBRID METHOD FOR NONLINEAR SYSTEMS*

MARIA GRAZIA GASPARO[†]

*Dipartimento di Energetica "S. Stecco,", Università di Firenze, Via C.Lombroso 6/17, 50134 Firenze, Italia*

*(Received 4 May 1998; In final form 19 May 1999)*

A nonmonotone hybrid method for solving nonlinear systems is proposed. The idea consists in matching a Newton-like method stabilized by nonmonotone line-search with a cheap direct search method that reduces monotonically the merit function. The aim is to enlarge the convergence domain of the Newton-like method without either losing its local convergence rate or introducing too much additional computational work. The proposed method has been extensively tested on a number of problems and compared with well assessed methods for nonlinear systems. The results suggest that the potential of the proposed approach is sometimes considerable.

*Keywords*: Nonlinear systems; nonmonotone line-search; hybrid methods; global convergence

## INTRODUCTION

Given a function $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$, we consider the numerical solution of the nonlinear system

$$F(x) = 0 \tag{1}$$

by Newton-like methods described by

$$H_k d_k = -F(x_k) \tag{2}$$

$$x_{k+1} = x_k + \lambda_k d_k, \tag{3}$$

* Work supported by M.U.R.S.T., Rome, Italy

[†] E-mail: gasparo@de.unifi.it

where $H_k$ is the nonsingular iteration matrix, $d_k$ is the search direction, $\lambda_k$ is the stepsize along this direction and an initial guess $x_0$ of a solution $x^*$ of (1) is given. Commonly used criteria to define $\lambda_k$ are based on the sufficient decrease of a suitable merit function $M(x)$, such that $M(x) \geq 0$ for all $x \in \mathbb{R}^n$ and $M(x) = 0$ iff $F(x) = 0$. However, such criteria can sometimes suffer of two major problems. If $d_k$ is not a descent direction for $M(x)$ at $x_k$, then the determination of $\lambda_k$ can be unfeasible and in this case the algorithm breaks down. Second, the practical rate of convergence of the generated sequence can be considerably slowed down by the descent requirement and a "stagnation" phenomenon can be observed, particularly when $x_0$ is a poor approximation of the solution. In recent years, two different strategies have been suggested to overcome these difficulties and obtain more flexible algorithms with actually large convergence domain. Breakdown in the algorithm can be avoided by using hybrid methods, where at each iteration $x_{k+1}$ is computed either by the Newton-like method or by a different global method for the minimization of $M(x)$. Poor actual rate of convergence and stagnation phenomena can be overcome by using nonmonotone strategies for the determination of $\lambda_k$, where the merit functions are allowed to increase at some iterations. In particular, this can help to avoid convergence towards a local minimizer of $M(x)$ that is not a solution of (1). The aforementioned stabilization tools have been firstly developed and experimented in the field of unconstrained minimization (see, for example, [11,12,18,21]). Some attractive extensions to the field of nonlinear systems have been proposed for the hybrid and nonmonotone approaches separately (see, for example, [1,2,4,7]), but limited practical experience have been carried out since now for hybrid nonmonotone methods ([8,10]), as far as the author is aware. In this paper a nonmonotone hybrid method, denoted by $H$-method, is proposed. It combines a globally convergent monotone direct search method for the minimization of

$$M(x) = 0.5\|F(x)\|^2, \tag{4}$$

where $\|\cdot\|$ is the euclidean norm, with a Newton-like method. The used iteration matrix $H_k$ is a difference approximation of the Jacobian $J(x_k)$ of $F(x)$ at $x_k$ and the stepsize $\lambda_k$ is obtained by line-search techniques. Both monotone and nonmonotone line-search strategies are considered. In the monotone case, the method yields an improved version of the Switching-method given in [1]. It exhibits enhanced efficiency and robustness features whereas maintaining the main theoretical properties of the original method.

The paper is organized as follows. In Section 2 the $H$-method is described; global convergence with asymptotic quadratic rate is proved, both for the monotone and nonmonotone line-search strategies. The results of an extensive numerical experimentation are presented in section 3. The $H$-method has been tested on a number of problems of small to medium size and of different difficulty levels and compared with other well assessed methods for nonlinear systems. From the obtained results, the $H$-method seems to be the most robust, both in the monotone and nonmonotone version, with actual convergence domain larger than the other methods. Further, the introduction of a nonmonotone line-search permits to solve some difficult problems where the monotone methods exhibit poor practical convergence rate or fail to converge. From now on, $e_j$ will denote the $j$-th unit coordinate vector in $\mathbb{R}^n$ and $\|\cdot\|$ both the euclidean norm in $\mathbb{R}^n$ and the induced matrix norm.

## THE *H*-METHOD

The $H$-method is a hybrid method which combines Newton-like (*NL*) iterations endowed with a monotone or nonmonotone line-search strategy for computing $\lambda_k$, with direct search (*DS*) iterations performed by a globally convergent direct search method for the minimization of (4). We briefly recall that the direct search methods for unconstrained minimization problems are methods that neither compute nor explicitly approximate derivatives of the objective function. The search is performed by comparing the values of the objective function at suitable points and does not require to solve any linear systems of equations or other expensive subproblems ([5,23]). These methods are monotone methods, in that they generate a sequence of iterates $\{x_k\}$ such that the associated values of the objective function decrease monotonically.

The aim of combining *NL* with *DS* iterations is to take advantage of the good features of both methods without either losing the local convergence properties of the *NL* method or introducing too much additional computational work with respect to the *NL* method. In other words, the aim is to obtain a robust method with a large convergence domain and good practical convergence rate.

First, an *NL* iteration will be described, under the assumptions that $H_k^{-1}$ exists with $\|H_k^{-1}\| \leq \beta$ and $\beta$ independent of $k$, and that a stepsize $\lambda_k$ can be determined within a finite number of operations. The adopted line-search strategy consists of a simple bisection technique that first of all checks the acceptability of $\lambda_k = 1$. More sophisticated backtracking techniques could be used as well (see, for example, [6]). The acceptance criterion is expressed in terms of a reference value $R_k \geq M(x_k)$ for the merit function. The choice $R_k = M(x_k)$ corresponds to the usual monotone line-search strategy, whereas the choice $R_k > M(x_k)$, for example $R_k = M(x_{k-p})$ for some $p > 0$, yields a nonmonotone strategy.

## *NL* Iteration

```
0. Given $x_k$, $R_k$, $H_k$, $\theta \in (0,1)$, $B_\lambda$
1. Solve the linear system $H_k d_k = -F(x_k)$
2. Find the smallest integer $i \in \{0, \dots, B_\lambda\}$ such that
       $$M(x_k + 2^{-i} d_k) \leq (1 - 2^{-i}\theta)R_k \qquad (5)$$
3. Set $\lambda_k = 2^{-i}$ and $x_{k+1} = x_k + \lambda_k d_k$
4. End of iteration.
```

Now, an iteration of the chosen direct search method will be described. It is the simplest direct search method, also known as coordinate search method or local variation method ([17,22]), which moves along coordinate directions choosing at each iteration the direction that is deemed more likely to go towards the global minimum of $M(x)$.

## *DS* Iteration

```
0. Given $x_k$, $\varepsilon_k > 0$
1. Set $a_k = \text{argmin}\{M(x_k + \varepsilon_k e_j),\ j = 1, \dots, n\}$
2. If $M(a_k) < M(x_k)$, then
      Set $x_{k+1} = a_k$ and go to Step 4.
   else
      Set $a_k = \text{argmin}\{M(x_k - \varepsilon_k e_j),\ j = 1, \dots, n\}$
      If $M(a_k) < M(x_k)$, then
         Set $x_{k+1} = a_k$ and go to Step 4.
      endif
   endif
```

3. Set $\varepsilon_k = 0.5\varepsilon_k$ and go to Step 1
4. End of iteration.

We remark that the definitions of $a_k$ at Step 1 and Step 2 could be swapped. The important thing is to visit all $2n$ points $x_k \pm \varepsilon_k e_j$, $j = 1, \ldots, n$, before halfing $\varepsilon_k$.

A *DS* iteration computes the values of $M(x)$ at $n$ trial points around $x_k$ and additional $n$ values are computed only if decrease in $M(x)$ is not achieved. Hence, this method is less expensive than the direct search method used in [1], where $2n$ values of $M(x)$ are always computed at each iteration.

The $H$-method combines *DS* iterations with *NL* iterations where $H_k$ is defined as a forward or backward difference approximation of $J(x_k)$. The same points around $x_k$ are used both as trial points in the *DS* iteration and as auxiliary points to compute $H_k$ in the *NL* iteration. The method is designed in such a way that an *NL* iteration is performed whenever possible and a *DS* iteration is carried out only when the *NL* iteration fails to find a new iterate. At this regard, we point out that the *NL* iteration is successful when the iteration matrix $H_k$ is invertible and (5) is satisfied within the finite bound $B_\lambda$ for the number of $i$-steps in the inner loop in Step 2.

Several rules to define the reference values $R_k$, $k = 0, 1, \ldots$, can be used, as long as the conditions $R_0 = M(x_0)$ and $R_{k+1} \leq R_k$ for $k \geq 0$ are satisfied ([7,8]). In this paper, we will use

$$R_k = \max\{M(x_k), M(x_{k-1}), \ldots, M(x_{k-\min(k,q)})\} \tag{6}$$

where $q \geq 0$ is a given integer. The choices $q > 0$ and $q = 0$ correspond to a nonmonotone and to a monotone strategy respectively.

## *H*-method

```
0. Given $x_0$, $\varepsilon_0$, $\theta \in (0,1)$, $B_\lambda$
1. Compute $M(x_0)$ and set $R_0 = M(x_0)$
2. For $k = 0, 1, \ldots$ until convergence, do:
   2.1 Set $\rho = \varepsilon_k$ and *iflag* $= 1$
   2.2 For $j = 1, \ldots, n$, do
          Compute $F_j = F(x_k + \rho e_j)$ and $M_j = M(x_k + \rho e_j)$
          Compute $H_k e_j = (F_j - F(x_k))/\rho$
   2.3 Try the *NL* iteration
```

```
2.4 If Step 2.3 is successful, then
        Compute $\varepsilon_{k+1} = \min\{\varepsilon_k, \|x_{k+1} - x_k\|, \|F(x_{k+1})\|\}$
        Compute $R_{k+1}$ by (6)
    else
        Compute $a_k = \text{argmin}\{M_1, \dots, M_n\}$
        If $M(a_k) < M(x_k)$, then
            Set $x_{k+1} = a_k$, $\varepsilon_{k+1} = \varepsilon_k$
            compute $R_{k+1}$ by (6)
        else if *iflag* = 1 then
            set *iflag* = 0, $\rho = -\varepsilon_k$ and go to Step 2.2
        else
            set $\varepsilon_k = 0.5\varepsilon_k$ and go to Step 2.1
        endif
    endif
End do
```

We remark that at each iteration the $H$-method makes two attempts to perform an *NL* iteration, one using forward differences and one using backward differences, and two attempts to perform a *DS* iteration before proceeding to half $\varepsilon_k$. For this reason, it turns out to be more flexible than the Switching-method of [1], where only one attempt is allowed.

The convergence of the $H$-method can be stated under the following standard assumptions:

A1) The level set $C(x_0) = \{x \in \mathbb{R}^n : M(x) \leq M(x_0)\}$ is bounded and contained in a convex open set $D$;

A2) $C(x_0)$ contains a non-zero finite number of points $x^*$ such that $F(x^*) = 0$;

A3) $F(x)$ is continuously differentiable on $D$;

A4) $J^{-1}(x)$ exists for all $x \in D$.

Remark that the existence of $J^{-1}(x)$ on $D$ implies its continuity on $D$ and its boundedness on $C(x_0)$ due to A1) and A3). Further, the previous assumptions ensure that $M(x)$ is continuously differentiable on $D$ and that the critical points of $M(x)$ are solutions of (1).

THEOREM 2.1 *Under assumptions A1)–A4) the sequence $\{x_k\}$ generated by the $H$-method converges to a point $x^*$ such that $F(x^*) = 0$.*

*Proof* Suppose that the $H$-method has constructed an infinite sequence $\{x_k\}$ and that there exists $K > 0$ such that all iterations are $DS$ iterations for $k \geq K$. Then convergence is ensured by the properties of the direct search method which can be proved with minor changes as in Theorem 3.1 of [1]. Otherwise, let $x_{k(j)}$, $j = 1, 2, \ldots$, be the iterates produced by $NL$ iterations and $\alpha = 1 - 2^{-B_\lambda}$. Then, for each $j$ we have $(1 - \theta\lambda_{k(j)-1}) < \alpha < 1$. Further, let $x_{l(j)}$, with $l(j) < k(j)$, be the point such that $R_{k(j)-1} = M(x_{l(j)})$. The sequence $\{M(x_{l(j)})\}$ is nonincreasing and bounded between zero and $\|F(x_0)\|$, hence it admits a limit $L$. Let $s+1$ be the index of the last iterate before $x_{l(j)}$ produced by an $NL$ iteration (for sufficiently large $j$, such a value $s$ exists). Between $x_{l(j)}$ and $x_{s+1}$ there are, if any, only iterates produced by $DS$ iterations, where $M(x)$ decreases. Then we have

$$M(x_{l(j)}) < M(x_{l(j)-1}) < \cdots < M(x_{s+1}) < \alpha R_s.$$

Let $m > 0$ such that $R_s = M(x_{l(j-m)})$. It follows that

$$M(x_{l(j)}) < \alpha M(x_{l(j-m)})$$

and then the sequence $\{M(x_{l(j)})\}$ tends to zero since it admits a subsequence converging to zero. In conclusion, $\|F(x_{k(j)})\| \to 0$ for $j \to \infty$. At this point, we see that $\|x_{k+1} - x_k\| \to 0$ for $k \to \infty$ either because $\varepsilon_k \to 0$ or because $\|F(x_k)\| \to 0$ and $\|H_k^{-1}\|$ is bounded. This fact and the assumption A2) complete the proof. □

The next theorem states that, after an initial phase where $DS$ iterations can occur, all iterates generated by the $H$-method are obtained by $NL$ iterations. Further, after the initial phase, $\lambda_k = 1$ will be always chosen and $M(x_{k+1}) < M(x_k)$ even if the nonmonotone line-search is implemented. Finally, quadratic convergence rate is proved. In order to state these results, we will suppose $F(x)$ twice continuously differentiable in $D$.

THEOREM 2.2 *Let A1)–A4) hold and $F(x)$ be twice continuously differentiable in $D$. Let $\{x_k\}$ be a convergent sequence generated by the $H$-method. Then there exists $K > 0$ such that for $k \geq K$ all iterates are produced by NL iterations with $\lambda_k = 1$ and $M(x_{k+1}) < M(x_k)$. Further, $\{x_k\}$ converges quadratically.*

*Proof* The proof follows, with minor modifications, from the proof of the Theorem 3.3 in [1]. □

## NUMERICAL EXPERIMENTS

The $H$-method was implemented by a double precision FORTRAN 77 code, named HYB. Both monotone ($q = 0$) and nonmonotone ($q > 0$) line-search strategies are included in the code. The linear algebraic systems are solved by $LU$ factorization. The double precision version of the routines F07ADF and F07AEF of the NAG Fortran Library Mk 17 (corresponding to the routines DGETRF and DGETRS of LAPACK) is used. It is well known that a generally useful way to stabilize the Newton-like methods is to implement a step control consisting in a normalization of $d_k$ ([8,15]). HYB is provided with the step control $d_k = \min\{1, \beta/\|d_k\|\}d_k$, with $\beta = 10^3 \max\{1, \|x_0\|\}$. However, the step control turned out to be essential for achieving convergence only in a few experiments.

The numerical tests were performed on a Risc 6000 workstation 3CT with a machine precision $\simeq 10^{-16}$. In all experiments convergence was declared at $x_k$ whenever $\|F(x_k)\| \leq \sqrt{n} \times 10^{-5}$. Failure was declared in one of the following occurrences: the number of performed iterations is greater than 500, $\varepsilon_k$ is halfed more than three times during an iteration, $\varepsilon_k$ becomes less than $10^{-11}$. The last two situations may be considered as indicative of the fact that $x_k$ is a critical point for $M(x)$ but not necessarily a solution of (1).

The results of the $H$-method are sensitive to the choice of the data $\varepsilon_0$ and $\theta$ and to the maximum number $B_\lambda$ of bisections allowed in the line-search procedure. All results here given were obtained by using the triplet $(\varepsilon_0, \theta, B_\lambda) = (0.1, 0.025, 3)$ which appeared very efficient. Further, after some experimenting, the value $q = 3$ was chosen in (6) for the nonmonotone strategy. From now on, we will denote by HYB0 and HYB3 the algorithms implemented in HYB with $q = 0$ and $q = 3$, respectively.

To comparison aims, we used some commercial and public domain codes to solve all the considered test problems with the same convergence criteria used for HYB. The used codes are C05NDF of the NAG Fortran Library Mk 17 and TENSOLVE [4]. The first code implements the hybrid method of Powell and uses rank-1 Broyden updating to approximate Jacobians with forward differences restarting when unsatisfactory progress is detected. TENSOLVE is a software package for solving nonlinear systems and nonlinear least-squares problems by globalized tensor methods or standard methods. In particular, we used tensor method and Newton's method, both globalized by monotone line-search, to which we will refer as LST and LSN respectively.

The five algorithms HYB0, HYB3, C05NDF, LST and LSN were used to solve a set of 241 small to medium size test problems with different difficulty levels. The test problems come from 27 different nonlinear systems of variable dimension $n$, to which we will refer as $S1, S2, \ldots, S27$, each of them solved starting from several initial guesses $x_0$ for several dimensions $n$ ranging between 10 and 100. $S1, S2, \ldots, S27$ have been used by many authors ([1,2,3,8,10,13,14,16,19,20]) to test global methods for nonlinear systems and are described in a separate report together with the complete results of the numerical experiments [9]. The used initial points for each nonlinear system are: the standard initial guess $x_s$ quoted in literature, $10x_s$, $100x_s$ and, whenever possible, the null vector here denoted as *null*. Obvious exceptions have been made for $S21$ where $x_s$ is the null vector and for $S25$ where $\|F(100x_s)\|$ caused overflow and $50x_s$ was used instead of $100x_s$.

HYB0, HYB3, C05NDF, LST and LSN have been compared by the point of view of their robustness, that is of their ability to achieve convergence for a given nonlinear system starting from different initial points. The results are summarized in Table I where the absolute and percentual number of successes obtained by each algorithm is shown. We separated the number of successes on the basis of the initial guess and in the second column the number of test problems corresponding to each initial guess is indicated. We point out that there exist 16 test problems where all algorithms failed to converge. These failures concern problems coming

TABLE I Absolute and Percentual Number of Successes for each Algorithm

| | | *HYB0* | *HYB3* | *C05NDF* | *LST* | *LSN* |
|---|---|---|---|---|---|---|
| $x_s$ | 67 | 59 (88%) | 61 (91%) | 62 (93%) | 53 (79%) | 58 (87%) |
| $10x_s$ | 64 | 57 (89%) | 55 (86%) | 48 (75%) | 42 (66%) | 51 (80%) |
| $50x_s$ | 2 | 2 (100%) | 2 (100%) | 0 | 0 | 0 |
| $100x_s$ | 62 | 47 (76%) | 49 (79%) | 38 (61%) | 35 (56%) | 43 (69%) |
| *null* | 46 | 42 (91%) | 45 (98%) | 28 (61%) | 39 (85%) | 40 (87%) |
| Total | 241 | 207 (86%) | 212 (88%) | 176 (73%) | 169 (70%) | 192 (80%) |

from the systems $S20$ and $S23$, defined very difficult problems in [8]. In fact, no algorithm was able to converge starting from $x_0 = 10x_s$ and $x_0 = 100x_s$ for $S20$ and from $x_0 \neq null$ for $S23$.

The results shown in Table I tend to suggest that HYB is the most robust among the considered algorithms, both using monotone and non-monotone line-search. HYB behaves more or less as the other algorithms for $x_0 = x_s$, while it is more reliable than the others for $x_0 \neq x_s$. We think that the "hybridness" is an essential factor for this behavior. In fact, some switchings from $NL$ iterations to $DS$ iterations or vice versa are needed to achieve convergence in several test problems (45 for $q = 0$ and 38 for $q = 3$). At least in the 50% of such problems, the initial guess is different from $x_s$ and most of the other considered algorithms fail.

From Table I, HYB0 and HYB3 seem to be essentially equivalent from the point of view of the robustness. From the detailed results available in [9] it can be seen that HYB3 permitted to avoid some failures of HYB0 both on "easy" problems, where the other algorithms converged, and on "very difficult" test problems, where the other algorithms, sometimes with the exception of LSN, failed. In order to give better insight into the relative performances of HYB0 and HYB3, let us consider the 195 problems where both converged within 100 iterations. For this set of problems, the mean number of iterations was 12 for HYB0 and 15 for HYB3 with a mean number of calls of the $LU$ factorization procedure equal to 15 and 17, respectively. Let us note that each call of the $LU$ factorization implies the computation of $n$ function values to construct a new matrix $H_k$. We deduce that the monotone version is a bit more efficient than the nonmonotone one and that one might loose in performance by using the nonmonotone method. This loss might be particularly important for large $n$ and/or expensive functions. On the other hand, the detailed results show that HYB0 is more efficient than HYB3 only for problems coming from the four systems $S5$, $S6$, $S7$ and $S8$.

The nonmonotone line-search is the winning strategy, both in terms of robustness and efficiency, for nonlinear systems such that the merit function features deep narrow curved valleys. Typically, once arrived near the bottom of such a valley, a monotone method generates a sequence of iterates which follows the valley's floor closely. This usually requires many iterations where the merit function decreases very slowly before reaching the region near the solution where the monotonicity becomes "natural." The initial phase can be so long that practically convergence is lost. The nonlinear systems $S\,10$, $S\,19$ and $S\,22$, described below, exhibit such features.

System $S\,10$ — **Extended Rosenbrock function** ([1,8,13,14,16,19])
For $i = 1, \ldots, n/2$ ($n$ even):

$$\begin{cases} F_{2i-1}(x) = 10(x_{2i} - x_{2i-1}^2) \\ F_{2i}(x) = 1 - x_{2i-1}. \end{cases}$$

This system admits the solution

$$x^* = (1.0, 1.0, 1.0, 1.0, \ldots)$$

and the standard initial guess is

$$x_s = (-1.2, 1.0, -1.2, 1.0, \ldots).$$

System $S\,19$ — **Augmented Powell badly scaled function** [8]
For $i = 1, \ldots, n/3$ ($n$ multiple of 3):

$$\begin{cases} F_{3i-2}(x) = 10^4 x_{3i-2} x_{3i-1} - 1. \\ F_{3i-1}(x) = \exp(-x_{3i-2}) + \exp(-x_{3i-1}) - 1.0001 \\ F_{3i}(x) = \phi(x_{3i}) \end{cases}$$

where

$$\phi(t) = \begin{cases} 0.5t - 2 & \text{if } t \le -1 \\ (-1924 + 4551t + 888t^2 - 592t^3)/1998 & \text{if } -1 < t < 2 \\ 0.5t + 2 & \text{if } t \ge 2. \end{cases}$$

This system admits the following solution, given with six correct digits,

$$x^* = (0.109816(-4), 9.10615, 0.399881, 0.109816(-4), \\ 9.10615, 0.399881, \ldots)$$

and the standard initial guess is

$$x_s = (0.0, 1.0, -4.0, 0.0, 1.0, -4.0, \ldots).$$

System $S\,22$ — **Diagonal of three variables function premultiplied by a quasi-orthogonal matrix** [8]
For $i = 1, \ldots, n/3$ ($n$ multiple of 3):

$$\begin{cases} F_{3i-2}(x) & = 0.6x_{3i-2} + 1.6x_{3i-1}^3 - 7.2x_{3i-1}^2 + 9.6x_{3i-1} - 4.8 \\ F_{3i-1}(x) & = 0.48x_{3i-2} - 0.72x_{3i-1}^3 + 3.24x_{3i-1}^2 - 4.32x_{3i-1} \\ & \quad -x_{3i} + 0.2x_{3i}^3 + 2.16 \\ F_{3i}(x) & = 1.25x_{3i} - 0.25x_{3i}^3. \end{cases}$$

This system admits the following solution, given with six correct digits,

$$x^* = (-0.231825(-14), 2.67765, 0.0, -0.231825(-14), 2.67765, 0.0, \ldots)$$

and the standard initial guess is

$$x_s = (50.0, 0.5, -1.0, 50.0, 0.5, -1.0, \ldots).$$

Several experiments were carried out using HYB0 and HYB3 to solve $S10$, $S19$ and $S22$ starting from several initial guesses of the form $x_0 = Cx_s$, with $C \in \mathbb{R}$. Some results of this set of experiments are shown in Tables II, III and IV, where the symbol *** is used to denote a failure. For each successful experiment the following data are reported:

NR = $\|F(\tilde{x})\|$, where $\tilde{x}$ is the last computed point;

IT = total number of iterations;

NF = total number of evaluations of $F(x)$;

NLU = total number of times the $LU$ factorization procedure was called;

NUP = total number of iterations where the value of the merit function actually increased (of course, only for $q = 3$).

Table II shows some results for $S10$ with $n = 100$. For different dimensions, the observed behavior is the same and the only changes

TABLE II Some Results for the Extended Rosenbrock Function ($n$ = 100)

| | $q=0$ | | | | $q=3$ | | | | |
|---|---|---|---|---|---|---|---|---|---|
| *C* | *NR* | *IT* | *NF* | *NLU* | *NR* | *IT* | *NF* | *NLU* | *NUP* |
| 0.0 | 0.0 | 8 | 821 | 8 | 0.0 | 6 | 613 | 6 | 1 |
| 0.1 | 0.0 | 8 | 820 | 8 | 0.0 | 5 | 511 | 5 | 2 |
| 0.3 | 0.0 | 8 | 820 | 8 | 0.0 | 5 | 511 | 5 | 2 |
| 0.5 | 0.0 | 10 | 1028 | 10 | 8.0(−15) | 7 | 717 | 7 | 2 |
| 0.7 | 0.0 | 12 | 1653 | 16 | 0.0 | 10 | 1031 | 10 | 3 |
| 0.9 | 0.0 | 9 | 1344 | 13 | 0.0 | 8 | 928 | 9 | 2 |
| 0.95 | 0.0 | 10 | 1343 | 13 | 0.0 | 8 | 927 | 9 | 2 |
| 1.0 | 0.0 | 9 | 1135 | 11 | 0.0 | 8 | 824 | 8 | 2 |
| 10.0 | 3.0(−13) | 3 | 305 | 3 | 3.0(−13) | 3 | 305 | 3 | 0 |
| 100.0 | 2.0(−11) | 3 | 305 | 3 | 2.0(−11) | 3 | 305 | 3 | 0 |

TABLE III Some Results for the Augmented Powell Badly Scaled Function ($n$ = 99)

| | $q = 0$ | | | | $q = 3$ | | | | |
|---|---|---|---|---|---|---|---|---|---|
| *C* | *NR* | *IT* | *NF* | *NLU* | *NR* | *IT* | *NF* | *NLU* | *NUP* |
| 0.0 | 4.0(−5) | 114 | 21013 | 212 | 7.0(−5) | 112 | 20808 | 210 | 4 |
| 1.0 | 5.0(−6) | 12 | 1203 | 12 | 5.0(−8) | 12 | 1202 | 12 | 0 |
| 2.0 | 2.0(−7) | 35 | 3581 | 35 | 9.0(−5) | 10 | 1002 | 10 | 1 |
| 4.0 | 4.0(−6) | 9 | 903 | 9 | 8.0(−5) | 8 | 802 | 8 | 1 |
| 6.0 | 8.0(−5) | 6 | 602 | 6 | 7.0(−5) | 80 | 802 | 8 | 0 |
| 10.0 | 7.0(−8) | 5 | 502 | 5 | 3.0(−8) | 8 | 802 | 8 | 0 |
| 14.0 | 8.0(−8) | 24 | 2424 | 24 | 2.0(−8) | 17 | 1708 | 17 | 3 |
| 20.0 | * * * | | | | * * * | | | | |
| 100.0 | * * * | | | | * * * | | | | |
| −1.0 | 1.0(−7) | 20 | 2015 | 20 | 5.0(−6) | 15 | 1506 | 15 | 2 |
| −2.0 | 8.0(−8) | 82 | 8403 | 82 | 6.0(−6) | 81 | 8302 | 81 | 2 |
| −4.0 | * * * | | | | 4.0(−6) | 19 | 1910 | 19 | 3 |
| −10.0 | * * * | | | | 6.0(−7) | 25 | 2509 | 25 | 3 |
| −20.0 | 5.0(−7) | 50 | 5047 | 50 | 3.0(−6) | 34 | 3406 | 34 | 3 |
| −40.0 | 5.0(−7) | 70 | 7047 | 70 | 8.0(−5) | 53 | 5305 | 53 | 3 |
| −60.0 | 2.0(−7) | 80 | 8018 | 80 | 9.0(−5) | 73 | 7304 | 73 | 3 |
| −80.0 | 5.0(−8) | 96 | 9609 | 96 | 8.0(−5) | 93 | 9304 | 93 | 4 |
| −100.0 | 7.0(−8) | 121 | 12119 | 121 | 8.0(−5) | 113 | 11304 | 113 | 4 |

TABLE IV Some Results for the Diagonal of Three Variables Function Premultiplied by a Quasi-orthogonal Matrix ($n$ = 99)

| | $q = 0$ | | | | $q = 3$ | | | | |
|---|---|---|---|---|---|---|---|---|---|
| C | NR | IT | NF | NLU | NR | IT | NF | NLU | NUP |
| 0.0 | * * * | | | | * * * | | | | |
| 1.0 | * * * | | | | 4.0(−6) | 6 | 602 | 6 | 1 |
| 10.0 | 4.0(−5) | 8 | 801 | 8 | 4.0(−5) | 8 | 801 | 8 | 0 |
| 100.0 | 3.0(−6) | 14 | 1401 | 14 | 3.0(−6) | 14 | 1401 | 14 | 0 |
| −1.0 | 2.0(−7) | 141 | 14503 | 141 | 1.0(−6) | 17 | 1703 | 17 | 2 |
| −4.0 | 5.0(−5) | 8 | 807 | 8 | 5.0(−5) | 10 | 1001 | 10 | 2 |
| −10.0 | * * * | | | | 6.0(−6) | 30 | 3006 | 30 | 8 |
| −20.0 | 6.0(−8) | 145 | 14903 | 145 | 5.0(−5) | 14 | 1402 | 14 | 2 |
| −30.0 | * * * | | | | 5.0(−5) | 13 | 1302 | 13 | 1 |
| −40.0 | 2.0(−6) | 16 | 1603 | 16 | 3.0(−6) | 24 | 2405 | 24 | 6 |
| −50.0 | 5.0(−6) | 15 | 1506 | 15 | 1.0(−6) | 18 | 1805 | 18 | 1 |
| −60.0 | * * * | | | | 5.0(−6) | 20 | 2003 | 20 | 3 |
| −70.0 | * * * | | | | 3.0(−7) | 15 | 1503 | 15 | 1 |
| −80.0 | * * * | | | | 7.0(−5) | 25 | 2501 | 25 | 7 |
| −90.0 | * * * | | | | * * * | | | | |
| −100.0 | * * * | | | | 3.0(−5) | 18 | 1803 | 18 | 1 |

occurs in NF since at each iteration at least $n + 1$ function evaluations are required whatever $q$ is. We remark that IT and NF for $q = 0$ are always greater or equal than the corresponding values for $q = 3$. Moreover, for

$q = 0$ the number NLU is often greater than IT. This means that at some iterations several attempts with different iteration matrices must be carried out to satisfy the decrease requirement.

The results concerning $S19$ are summarized in Table III, where $n = 99$ is considered. Also for this system the convergence behavior is qualitatively the same for each $n$, except for a few isolated cases. From the table, we see that for some initial points the monotone strategy is more efficient than the nonmonotone one, but the opposite is true for those initial guesses which require a high number of iterations.

Finally, Table IV shows some results obtained for $S22$ with $n = 99$. For this system the monotone and nonmonotone strategies exhibit generally the same behavior for $x_0 = Cx_s$ with $C \geq 0$, whereas they behave very differently for $C < 0$. We observe that the nonmonotone strategy is less efficient than the monotone one for some initial guesses ($C = -4$, $C = -40$ and $C = -50$). On the other hand, several failures of the monotone strategy are solved by using $q = 3$.

## CONCLUSIONS AND PERSPECTIVES

The computational results presented in Section 3 indicate that the combination with a slow and steady method such as a direct search method can be a very good tool to stabilize traditional Newton-like methods. Further, the introduction of nonmonotone line-search strategies allows to solve efficiently very difficult problems where traditional monotone methods fail to converge. So, nonmonotone hybrid direct search-Newton like methods seem to be preferable to many traditional methods for solving general systems of nonlinear equations.

The author believes that further investigations of the described approach may be interesting, both in the direction of nonmonotonicity and in the direction of the hybridness. Other possible rules to define the reference values $R_k$ can be introduced and investigated and more general direct search methods can be used. In particular, we are thinking about direct search methods endowed with suited step control mechanisms and reflections, expansions, contractions techniques, in such a way that the convergence of the overall hybrid method can be accelerated.

The approach proposed in this paper could be applied to nonlinear least squares problems, in order to enlarge the domain of convergence of the Gauss-Newton method. It is known that this method uses analytic or finite difference approximated Jacobian matrices, so a matching with direct search methods can be efficiently carried out. Further, nonmonotone line-search strategies can be fruitfully incorporated in the Gauss-Newton method instead of the usually considered monotone line-search.

We are currently investigating the possibility of extending our nonmonotone hybrid approach to tensor methods for nonlinear systems and least squares problems, in such a way to deal also problems where the Jacobian is singular or ill conditioned at the solution or at some iterates.

## *References*

[1] Bellavia, S., Gasparo, M. G. and Macconi, M. (1996). A Switching-Method for nonlinear systems. *J. of Computational and Applied Mathematics*, **71**, 83–93.

[2] Bellavia, S., Gasparo, M. G. and Macconi, M. (1996). Partially updated Switching-Method for systems of nonlinear equations. *J. of Computational and Applied Mathematics*, **76**, 77–88.

[3] Bogle, I. D. L. and Perkins, J. D. (1990). A new sparsity preserving quasi-Newton update for solving nonlinear equations. *SIAM J. Sci. Stat. Comput.*, **11**, 621–630.

[4] Bouaricha, A. and Schnabel, R. B. (1997). Algorithm 768: TENSOLVE: A Software Package for Solving Systems of Nonlinear Equations and Nonlinear Least-Squares Problems Using Tensor Methods. *ACM Transactions on Mathematical Software*, **23**, 174–195.

[5] Conn, A. R. and Scheinberg, K. Ph.L. (1997). Toint Recent progress in unconstrained nonlinear optimization without derivatives. *Math. Programming*, **79**, 397,414.

[6] Dennis, J. E., Jr. and Schnabel, R. B. (1983). *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ.

[7] Ferris, M. C. and Lucidi, S. (1994). Nonmonotone stabilization methods for nonlinear equations. *J. of Optimization Theory and Applications*, **81**, 53–71.

[8] Friedlander, A., Gomes-Ruggiero, M. A., Kozakevich, D. N., Martinez, J. M. and Santos, S. A. (1997). Solving nonlinear systems of equations by means quasi-Newton methods with a nonmonotone strategy. *Optimization Methods and Software*, **8**, 25–51.

[9] Gasparo, M. G. (1998). *A nonmonotone hybrid method for nonlinear systems: the complete numerical results*, Technical report 98/5, Dipartimento di Energetica, Universita' di Firenze, Italia.

[10] Gomez-Ruggiero, M. A., Martinez, J. M. and Moretti, A. C. (1992). Comparing algorithms for solving sparse nonlinear systems of equations. *SIAM J. on Sci. Stat. Comput.*, **13**, 459–483.

[11] Grippo, L., Lampariello, F. and Lucidi, S. (1986). A nonmonotone line search technique for Newton's method. *SIAM J. Numer. Anal.*, **23**, 707–716.

[12] Grippo, L., Lampariello, F. and Lucidi, S. (1991). A class of nonmonotone Stabilization Methods in Unconstrained Optimization. *Numerische Mathematik*, **59**, 779–805.

[13] Li, G. (1989). Successive column correction algorithms for solving sparse nonlinear systems of equations. *Math. Programming*, **43**, 187–207.

[14] Luksan, L. (1994). Inexact Trust Region Method for Large Sparse Systems of Nonlinear Equations. *J. on Opt. Theory and Appl.*, **81**, 569–590.

[15] Moré, J. J. and Cosnard, M. Y. (1979). Numerical solution of nonlinear equations. *ACM Trans. on Math. Software*, **5**, 64–85.

[16] Moré, J. J., Garbow, B. S. and Hillstrom, K. E. (1981). Testing Unconstrained Optimization Software. *ACM Trans. on Math. Software*, **7**, 17–41.
[17] Polak, P. (1971). *Computational Methods in Optimization: A Unified Approach*. Ac. Press, N.Y.
[18] Polak, E. (1976). On the Global Stabilization of Locally Convergent Algorithms. *Automatica*, **12**, 337–342.
[19] Spedicato, E. and Huang, Z. (1996). Numerical experience with Newton-like methods for nonlinear algebraic systems. *Tech. Rep. Dip. M.S.I. Appl.*, **15**.
[20] Toint, Ph. L. (1986). Numerical Solution of Large Sets of Algebraic Nonlinear Equations. *Math. of Comput.*, **46**, 175–189.
[21] Toint, Ph. L. (1996). An assessment of non-monotone line-search techniques for unconstrained minimization. *SIAM J. on Sci. Comput.*, **17**, 725–739.
[22] Torczon, V. (1997). On the convergence of pattern search algorithms. *SIAM J. Optim.*, **7**, 1–25.
[23] Wright, M. H. (1996). *Direct search methods: Once scorned, now respectable* in: D. F. Griffiths and G. A. Watson eds *Proceedings of the 1995 Dundee Biennal Conference in Numerical Analysis* Addison-Wesley, reading, MA and Longman, Harlow, UK.