A Nonmonotone Line Search Method for Symmetric Nonlinear Equations*

Gonglin Yuan¹, Laisheng Yu²

¹College of Mathematics and Information Science, Guangxi University, Nanning, China
²Student Affairs Office, Guangxi University, Nanning, China
E-mail: glyuan@gxu.edu.cn
Received February 7, 2010; revised March 21, 2010; accepted July 7, 2010

Abstract

In this paper, we propose a new method which based on the nonmonotone line search technique for solving symmetric nonlinear equations. The method can ensure that the search direction is descent for the norm function. Under suitable conditions, the global convergence of the method is proved. Numerical results show that the presented method is practicable for the test problems.

Keywords: Nonmonotone Line Search, Symmetric Equations, Global Convergence

1. Introduction

Consider the following nonlinear equations:

\[ g(x) = 0, x \in \mathbb{R}^n \]  \hspace{1cm} (1)

where \( g: \mathbb{R}^n \rightarrow \mathbb{R}^n \) be continuously differentiable and its Jacobian \( \nabla g(x) \) is symmetric for all \( x \in \mathbb{R}^n \). This problem can come from unconstrained optimization problems, a saddle point problem, and equality constrained problems [the detail see [1]]. Let \( \phi(x) \) be the norm function defined by \( \phi(x) = \frac{1}{2} \| g(x) \|^2 \). Then the nonlinear equation problem (1) is equivalent to the following global optimization problem

\[ \min \phi(x), \ x \in \mathbb{R}^n \]  \hspace{1cm} (2)

The following iterative formula is often used to solve (1) and (2):

\[ x_{k+1} = x_k + \alpha_k d_k \]

where \( \alpha_k \) is a steplength, and \( d_k \) is one search direction. To begin with, we briefly review some methods for (1) and (2) by line search technique. First, we give some techniques for \( \alpha_k \). Li and Fukushima [1] proposed an approximate monotone line search technique to obtain the step-size \( \alpha_k \) satisfying:

\[ \varphi(x_k + \alpha_k d_k) - \varphi(x_k) \leq -\delta_1 \| \alpha_k d_k \|^2 - \delta_2 \| \alpha_k g_k \|^2 + \varepsilon_k \| g_k \|^2 \]  \hspace{1cm} (3)

where \( \delta_1 > 0 \) and \( \delta_2 > 0 \) are positive constants, \( \alpha_k = r^j_k \), \( r \in (0,1), l_k \) is the smallest nonnegative integer \( i \) such that (3), and \( \varepsilon_k \) satisfies \( \sum_{k=0}^{\infty} \varepsilon_k < \infty \). Combining the line search (3) with one special BFGS update formula, they got some better results (see [1]). Inspired by their idea, Wei [2] and Yuan [3] made a further study. Brown and Saad [4] proposed the following line search method to obtain the stepsize

\[ \varphi(x_k + \alpha_k d_k) - \varphi(x_k) \leq \sigma \alpha_k \nabla \varphi(x_k)^T d_k \]  \hspace{1cm} (4)

where \( \sigma \in (0,1) \). Based on this technique, Zhu [5] gave the nonmonotone line search technique:

\[ \varphi(x_k + \alpha_k d_k) - \varphi(x_k) \leq \sigma \alpha_k \nabla \varphi(x_k)^T d_k \]  \hspace{1cm} (5)

where \( \varphi_k = \max_{0 \leq j \leq n(k)} \{ q_{k-j} \}, k = 0,1,2,\ldots, n(k) = \min \{ M, k \} \), and \( M \geq 0 \) is an integer constant. From these two techniques (4) and (5), it is easy to see that the Jacobian matrix \( \nabla g(x) \) must be computed at every iteration, which will increase the workload especially for large-scale problems or this matrix is expensive. Considering these points, we [6] presented a new backtracking inexact technique to obtain the stepsize \( \alpha_k \).

*This work is supported by China NSF grands 10761001, the Scientific Research Foundation of Guangxi University (Grant No. X081082), and Guangxi SF grands 0991028.
\[ \| g(x_k + \alpha_k d_k) \| \leq \| g(x_k) \| + \delta \alpha_k^2 q(x_k)^T d_k \]  
(6)

where \( \delta \in (0, 1) \). Second, we present some techniques for \( d_k \). One of the most effective methods is Newton method. It normally requires a fewest number of function evaluations, and it is very good at handling ill-conditioning. However, its efficiency largely depends on the possibility to efficiently solve a linear system which arises when computing the search \( d_k \) at each iteration

\[ \nabla g(x_k) d_k = -g(x_k) \]  
(7)

Moreover, the exact solution of the system (7) could be too burdensome, or it is not necessary when \( d_k \) is far from a solution [7]. Inexact Newton methods [5, 7] represent the basic approach underlying most of the Newton-type large-scale algorithms. At each iteration, the current estimate of the solution is updated by approximately solving the linear system (7) using an iterative algorithm. The inner iteration is typically “truncated” before the solution to the linear system is obtained. Griewank [8] firstly proposed the Broyden’s rank one technique which can ensure the descent search direction on the norm function for solving symmetric nonlinear Equations (1) and prove the global convergence of our method. The numerical results are reported too. Here and throughout this paper, \( ||.|| \) denote the Euclidian norm of vectors or its induced matrix norm.

This paper is organized as follows. In the next section, we will give our algorithm for (2). The global convergence and the numerical result are established in Section 3 and in Section 4, respectively.

2. The Algorithm

Precisely, our algorithm is stated as follows.

Algorithm 1.

Step 0: Choose an initial point \( x_0 \in \mathbb{R}^n \), an initial symmetric positive definite matrix \( B_0 \in \mathbb{R}^{n \times n} \), and constants \( r \in (0, 1) \), \( 0 \leq \rho_1 \leq \rho_2 \leq 1, 0 < \delta_1 < 1, J_0 = \| g(x_0) \|^2, E_0 = 1, \) and \( k = 0; \)

Step 1: If \( g_k = 0; \) then stop; Otherwise, solving the following linear Equations (10) to obtain \( d_k \) and go to step 2;

\[ B_k d_k + g_k = 0 \]  
(10)

Step 2: Let \( i_k \) be the smallest nonnegative integer \( i \) such that

\[ \| g(x_k + \lambda_k d_k) \| \leq \| g(x_k) \| + \delta \lambda_k^2 q(x_k)^T d_k \]  
(11)

holds for \( \lambda = \rho^i \). Let \( \lambda_k = \rho^{i_k} \);

Step 3: Let \( x_{k+1} = x_k + \alpha_k d_k \), \( s_k = x_{k+1} - x_k \) and \( y_k = g(x_{k+1}) - g(x_k) \). If \( y_k^T s_k > 0 \), update \( B_k \) to generate \( B_{k+1} \) by the BFGS formula (9). Otherwise, let \( B_{k+1} = B_k; \)

Step 4: Choose \( \rho_k \in [0,1], \) and set

\[ E_{k+1} = \rho_k E_k + 1, J_{k+1} = \frac{\rho_k E_k f(s_k) + \| g(x_{k+1}) \|^2}{E_{k+1}} \]  
(12)

Step 5: Let \( k = k + 1, \) go to Step 1.

Remark 1: 1) By the technique of the step 3 in the algorithm [see [1]], we deduce that \( B_{k+1} \) can inherits the positive and symmetric properties of \( B_k \). Then, it is not difficult to get \( d_k^T g_k < 0 \).
2) It is easy to know that $J_{k+1}$ is a convex combination of $J_k$ and $\|g(x_{k+1})\|^2$. By $J_0 = \|g_0\|^2$, it follows that $J_k$ is a convex combination of function values $\|g_0\|^2, \|g_1\|^2, \ldots, \|g_k\|^2$. The choice of $\rho_k$ controls the degree of nonmonotonicity. If $\rho_k = 0$ for each $k$, then the line search is the usual monotone line search. If $\rho_k = 1$ for each $k$, then $V_k = \frac{1}{k+1} \sum_{i=0}^{k+1} \|g_i\|^2$, where is the average function value.

3) By (9), we have $B_{k+1} = B_k = g_{k+1} - g_k = \nabla g_{k+1} s_k = \nabla g_{k+1} t_k$, this means that $B_{k+1}$ approximate to $\nabla g_{k+1}$ along $s_k$.

3. The Global Convergence Analysis of Algorithm 1

In this section, we establish global convergence for Algorithm 1. The level set $\Omega$ is defined by

$$\Omega = \left\{ x \in \mathbb{R}^n \mid \|g(x)\| \leq \|g(x_0)\| \right\}.$$  

**Assumption A.** The Jacobian of $g$ is symmetric and there exists a constant $M > 0$ holds

$$\|g(x) - g(x_k)\| \leq M \|x - x_k\|$$  

for $x \in \Omega$.

Since $B_k$ approximates $\nabla g_k$ along direction $s_k$, we can give the following assumption.

**Assumption B.** $B_k$ is a good approximation to $\nabla g_k$, i.e.,

$$\|\nabla g(x_k) - B_k\| d_k \leq \varepsilon \|s_k\|$$  

where $\varepsilon \in (0,1)$ is a small quantity.

**Assumption C.** There exist positive constants $b_1$ and $b_2$ satisfy

$$g_k^T d_k \leq -b_1 \|g_k\|^2$$  

and

$$\|d_k\| \leq b_2 \|s_k\|$$

for all sufficiently large $k$.

By (10) and Assumption C, we have

$$b_1 \|s_k\| \leq \|d_k\| \leq b_2 \|s_k\|$$  

**Lemma 3.1.** Let Assumption B hold and $\{\alpha_k, d_k, x_{k+1}, g_{k+1}\}$ be generated by Algorithm 1. Then $d_k$ is descent direction for $\phi(x)$ at $x_k$, i.e.,

$$\nabla \phi(x_k)^T d_k < 0$$

**Proof.** By (10), we have

$$\nabla \phi(x_k)^T d_k = g_k^T \nabla g_k d_k = g_k^T (\nabla g_k d_k - B_k) d_k - g_k^T g_k$$

$$= g_k^T (\nabla g_k d_k - B_k) d_k - g_k^T g_k$$  

Using (14) and taking the norm in the right-hand-side of (19), we get

$$\nabla \phi(x_k)^T d_k \leq \|g_k^T (\nabla g_k d_k - B_k) d_k\| - \|g_k\|^2$$

$$\leq -(1-\varepsilon) \|g_k\|^2$$

Therefore, for $\varepsilon \in (0,1)$, we get the lemma.

By the above lemma, we know that the norm function $\phi(x)$ is descent along $d_k$, then $\|g_k\| \leq \|g_k\|$. Hence, we have for all $k$

$$\|g_k\| \leq \|g_k\| \leq \cdots \leq \|g_0\|$$

Which means that $\{x_k\} \subseteq \Omega$.

The next lemma will show that for any choice of $\rho_k \in [0,1], J_k$ lies between $\|g_k\|$ and $V_k$.

**Lemma 3.2.** Let Assumption B hold and $\{\alpha_k, d_k, x_{k+1}, g_{k+1}\}$ be generated by Algorithm 1. Then $\{x_k\} \subseteq \Omega$. Moreover, $\|g_k\|$ converges.

**Proof.** By Lemma 3.1, we get $\|g_{k+1}\| \leq \|g_k\|$. Then, we conclude that $\|g_k\|$ converges. Moreover, we have for all $k$

$$\|g_{k+1}\| \leq \|g_k\| \leq \cdots \leq \|g_0\|$$

The next lemma will show that for any choice of $\rho_k \in [0,1], J_k$ lies between $\|g_k\|$ and $V_k$.

**Lemma 3.3.** Let $\{\alpha_k, d_k, x_{k+1}, g_{k+1}\}$ be generated by Algorithm 1, we have $\|g_k\| \leq J_k \leq V_k$, $J_{k+1} \leq J_k$ for each $k$.

**Proof.** We will prove the lower bound for $J_k$ by induction. For $k = 0$, by the initialization $J_0 = \|g_0\|^2$, this holds. Now we assume that $J_i \geq \|g_i\|^2$ holds for all $0 \leq i \leq k$. By (2.3) and $\|g_i\|^2 \leq \|g_i\|^2$, we have

$$J_{k+1} = \rho_k E_i J_i + \|g_{k+1}\|^2 \geq \rho_k E_i \|g_i\|^2 + \|g_{k+1}\|^2$$

where $E_i = \rho_i E_i + 1$. Now we prove that $J_{k+1} \leq J_k$ is true. By (12), and using $\|g_i\|^2 \geq \|g_{k+1}\|^2$, we obtain

$$J_{k+1} = \rho_k E_i J_i + \|g_{k+1}\|^2 \leq \rho_k E_i J_i + J_{k+1}$$
Which means that $J_{k+1} \leq J_k$ for all $k$ is satisfied.

Then we have

$$\|g_k\|^2 \leq J_k \leq J_{k-1}$$  \hspace{1cm} (22)

Let $L_k : R \rightarrow R$ be defined by

$$L_k(t) = \frac{\nu J_{k-1} + \|g_k\|^2}{t+1},$$

we can get

$$L_k(t) = \frac{\nu J_{k-1} + \|g_k\|^2}{(t+1)^{\nu}}.$$  \hspace{1cm} (23)

By $J_{k-1} \leq \|g_k\|^2$, we obtain $L_k(t) \geq 0$ for all $t \geq 0$.

Then, $L_k$ is nondecreasing, and $\|g_k\|^2 = L_k(0) \leq L_k(t)$ for all $t \geq 0$. Now we prove the upper bound $J_k \leq V_k$ by induction. For $k = 0$, by the initialization $J_0 = \|g_0\|^2$, this holds. Now assume that $J_j \leq V_j$ hold for all $0 \leq j \leq k$. By using $E_0 = 1$, (12), and $\rho_k \in [0,1]$, we obtain

$$E_{j+1} = 1 + \sum_{i=0}^{j} \rho_{j-i} \leq j + 2$$  \hspace{1cm} (24)

Denote that $L_k$ is monotone nondecreasing, (23) implies that

$$J_k = L_k \left(\rho_{k-1} E_{k-1}\right) = L(0,1) \leq L_k(k)$$  \hspace{1cm} (25)

Using the induction step, we have

$$L_k(k) = \frac{k J_{k-1} + \|g_k\|^2}{k+1} \leq k V_{k-1} + \|g_k\|^2 = V_k$$  \hspace{1cm} (26)

Combining (24) and (25) implies the upper bound of $J_k$ in this lemma. Therefore, we get the result of this lemma.

The following lemma implies that the line search technique is well-defined.

**Lemma 3.4.** Let Assumption A, B and C hold. Then Algorithm 1 will produce an iterate $x_{k+1} = x_k + \alpha_k d_k$ in a finite number of backtracking steps.

**Proof.** From Lemma 3.8 in [4] we have that in a finite number of backtracking steps, $\alpha_k$ must satisfy

$$\|g(x_k + \alpha_k d_k)\|^2 - \|g(x_k)\|^2 \leq \sigma \alpha_k (1 - \varepsilon) \|g_k\|^2 \nabla g_k d_k$$  \hspace{1cm} (27)

where $\sigma \in (0,1)$. By (20) and (15), we get

$$\alpha_k \nabla g_k d_k \leq -\alpha_k (1 - \varepsilon) \|g_k\|^2$$

Using $\alpha_k \in [0,1]$, we obtain

$$\alpha_k \nabla g_k d_k \leq -\alpha_k (1 - \varepsilon) \|g_k\|^2 \nabla g_k d_k$$  \hspace{1cm} (28)

where $\alpha_k \in (0,1)$, we obtain

$$\alpha_k \nabla g_k d_k \leq -\alpha_k (1 - \varepsilon) \|g_k\|^2 \nabla g_k d_k$$  \hspace{1cm} (29)

**Proof.** Assuming the step-size $\alpha_k$ such that (11). Then $\alpha_k \alpha_k = \alpha_k$ does not satisfy (11), i.e.,

$$\|g(x_k + \alpha_k d_k)\|^2 - J_k > \delta \alpha_k^2 \|g_k\|^2 \nabla g_k d_k$$

By $\|g_k\|^2 \leq J_k$, we get

$$\|g(x_k + \alpha_k d_k)\|^2 - \|g_k\|^2 \leq J_k > \delta \alpha_k^2 \|g_k\|^2 \nabla g_k d_k$$

Which implies that

$$-\delta \alpha_k^2 \|g_k\|^2 \nabla g_k d_k > \|g(x_k + \alpha_k d_k)\|^2 - \|g_k\|^2$$  \hspace{1cm} (30)

By Taylor formula, (19), (20), and (17), we get

$$\|g(x_k + \alpha_k d_k)\|^2 - \|g_k\|^2 = -2 \alpha_k \nabla g_k d_k \nabla g_k d_k$$

$$+ O(\alpha_k^2 \|d_k\|^2) \geq \alpha_k(1 - \varepsilon) \|g_k\|^2 + O(\alpha_k^2 \|d_k\|^2)$$  \hspace{1cm} (31)

Using (15), (17), (30), and (31) we obtain

$$\alpha_k \nabla g_k d_k \leq \frac{2(1 - \varepsilon)}{b_2^2} \|d_k\|^2$$

$$\geq 2 \alpha_k^2 (1 - \varepsilon) \frac{1}{b_1^2} \|d_k\|^2 + O(\alpha_k^2 \|d_k\|^2)$$

Using $\alpha_k \in [0,1]$, we obtain

$$\alpha_k \nabla g_k d_k \leq \frac{2(1 - \varepsilon)}{b_2^2} \|d_k\|^2$$

$$\geq 2 \alpha_k^2 (1 - \varepsilon) \frac{1}{b_1^2} \|d_k\|^2 + O(\alpha_k^2 \|d_k\|^2)$$

Copyright © 2010 SciRes.
which implies when \( k \) sufficiently large,
\[
\alpha_k' \geq \frac{b_1 (1 - \varepsilon)}{2(1 - \varepsilon)b_1 + b_2 \delta_k'}.
\]

Let
\[
b_0 \in \left(0, \frac{b_1 (1 - \varepsilon)}{2(1 - \varepsilon)b_1 + b_2 \delta_k'}\right).
\]
The proof is complete.

In the following, we give the global convergence theorem.

**Theorem 3.1.** Let \( \{\alpha_k, d_k, x_{k+1}, g_{k+1}\} \) be generated by Algorithm 1, Assumption A, B, and C hold, and \( \|g_k\| \) be bounded from below. Then
\[
\lim \inf_{k \to \infty} \|g_k\| = 0 \quad (33)
\]
Moreover, if \( \rho_2 < 1 \), then
\[
\lim_{k \to \infty} \|g_k\| = 0 \quad (34)
\]
Therefore, every convergent subsequence approaches to a point \( x^* \), where \( g(x^*) = 0 \).

**Proof.** By (11), (15), (16), and (19), we have
\[
\|g_{k+1}\| \leq J_k + \delta_2 \alpha_k d_k^T g_k \leq J_k - \alpha_k \delta_1 \|g_k\|^2
\]
(35)
Let \( \varepsilon = \delta_1 \delta_2 \). Combining (12) and the upper bound of (35), we get
\[
J_{k+1} = \frac{\rho_1 E_k J_k + \|g_{k+1}\|^2}{E_{k+1}} \leq \frac{\rho_1 E_k J_k + J_k - \varepsilon \|g_k\|^2}{E_{k+1}}
\]
(36)
Since \( \|g_k\|^2 \) is bounded from below and \( \|g_k\|^2 \leq J_k \) for all \( k \), we can conclude that \( J_k \) is bounded from below. Then, using (36), we obtain
\[
\sum_{k=0}^{\infty} \frac{\|g_k\|^2}{E_{k+1}} < \infty \quad (37)
\]
By (23), we get
\[
E_{k+1} < k + 2 \quad (38)
\]
If \( \|g_k\|^2 \) were bounded away from 0, then (37) would violate (38). Hence, (33) holds. If \( \rho_2 < 1 \), by (23), we have
\[
E_{k+1} = 1 + \sum_{i=0}^{k} \rho_{k-i} \leq 1 + \sum_{j=0}^{k} \rho_{j} \rho_{j+1} \leq \sum_{j=0}^{k} \rho_{2} = \frac{1}{1 - \rho_2}
\]
(39)
Then, (37) implies (34). The proof is complete.

### 4. Numerical Results

In this section, we report the results of some numerical experiments with the proposed method.

**Problem 1.** The discretized two-point boundary value problem is the same to the problem in [22]
\[
g(x) = Ax + \frac{1}{(n+1)^2} F(x),
\]
where \( A \) is the \( n \times n \) tridiagonal matrix given by
\[
A = \begin{bmatrix}
4 & -1 & & & \\
-1 & 4 & -1 & & \\
& \ddots & \ddots & \ddots & \\
& & -1 & 4 & -1
\end{bmatrix}
\]
and \( F(x) = (F_1(x), F_2(x), \ldots, F_n(x))^T \), with \( F_i(x) = \sin x_i - 1, i = 1, 2, \ldots, n \).

**Problem 2.** Unconstrained optimization problem
\[
\min_{x \in \mathbb{R}^n} f(x),
\]
with Enval function [23] defined by
\[
f(x) = \sum_{i=2}^{n} \left( x_i^2 + x_{i-1}^2 \right)^2 - 4x_{i-1} + 3
\]
The related symmetric nonlinear equation is
\[
g(x) = \frac{1}{4} \nabla f(x)
\]
where \( g(x) = (g_1(x), g_2(x), \ldots, g_n(x))^T \) with
\[
\begin{align*}
g_1(x) &= x_i \left( x_i^2 + x_{i-1}^2 \right) - 1 \\
g_i(x) &= x_i \left( x_i^2 + x_{i-1}^2 \right) - 1, i = 2, 3, \ldots, n - 1 \\
g_n(x) &= x_n \left( x_n^2 + x_{n-1}^2 \right)
\end{align*}
\]
In the experiments, the parameters in Algorithm 1 were chosen as \( r = 0.1, \delta_1 = 0.001, \rho_2 = 0.8, B_0 \) is unit matrix. The program was coded in MATLAB 6.1. We stopped the iteration when the condition \( \|g(x)\| < 10^{-6} \) was satisfied. Tables 1 and 2 show the performance of the method need to solve the Problem 1. Tables 3 and 4 show the performance of the method need to solve the Problem 2. The columns of the tables have the following meaning:

- **Dim:** the dimension of the problem.
- **NI:** the total number of iterations.
- **NG:** the number of the function evaluations.
- **GG:** the function evaluations.

From the above tabulars, we can see that the numerical
Test Result for the Problem 1

Table 1. Small-scale.

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>(4, ..., 4)</th>
<th>(20, ..., 20)</th>
<th>(100, ..., 100)</th>
<th>(-4, ..., -4)</th>
<th>(-20, ..., -20)</th>
<th>(-100, ..., -100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dim</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
</tr>
<tr>
<td>$n = 10$</td>
<td>16/22/3.714814e-7</td>
<td>16/20/2.650667e-7</td>
<td>16/20/4.014933e-7</td>
<td>16/22/3.723721e-7</td>
<td>16/20/2.643713e-7</td>
<td>16/20/4.013770e-7</td>
</tr>
<tr>
<td>$n = 50$</td>
<td>44/47/1.388672e-7</td>
<td>44/49/6.929935e-7</td>
<td>46/49/3.713174e-7</td>
<td>44/47/1.388793e-7</td>
<td>44/47/6.929516e-7</td>
<td>46/49/3.726373e-8</td>
</tr>
<tr>
<td>$n = 100$</td>
<td>68/71/5.905592e-7</td>
<td>70/73/8.759459e-7</td>
<td>72/75/3.125373e-7</td>
<td>68/71/5.905724e-7</td>
<td>70/73/8.759500e-7</td>
<td>72/75/3.125382e-7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>(4, 0, ..., 0)</th>
<th>(20, 0, ..., 20)</th>
<th>(100, 0, ..., 100)</th>
<th>(-4, 0, ..., -4)</th>
<th>(-20, 0, ..., -20)</th>
<th>(-100, 0, ..., -100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dim</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
</tr>
<tr>
<td>$n = 100$</td>
<td>65/67/6.046233e-7</td>
<td>69/71/7.845591e-7</td>
<td>1/73/6.960858e-7</td>
<td>65/67/6.046254e-7</td>
<td>9/71/7.845692e-7</td>
<td>71/73/6.996092e-7</td>
</tr>
</tbody>
</table>

Table 2. Large-scale.

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>(4, ..., 4)</th>
<th>(20, ..., 20)</th>
<th>(30, ..., 30)</th>
<th>(-4, ..., -4)</th>
<th>(-20, ..., -20)</th>
<th>(-30, ..., -30)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dim</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
</tr>
<tr>
<td>$n = 300$</td>
<td>70/73/7.844778e-7</td>
<td>76/79/7.741702e-7</td>
<td>8/73/6.759628e-7</td>
<td>0/73/7.844800e-7</td>
<td>6/79/7.741706e-7</td>
<td>78/81/6.759631e-7</td>
</tr>
<tr>
<td>$n = 500$</td>
<td>70/73/8.547195e-7</td>
<td>76/79/8.435874e-7</td>
<td>8/73/8.547204e-7</td>
<td>0/73/8.435876e-7</td>
<td>78/81/7.366073e-7</td>
<td>78/81/7.366073e-7</td>
</tr>
<tr>
<td>$n = 800$</td>
<td>68/70/6.505423e-7</td>
<td>74/76/6.414077e-7</td>
<td>4/76/9.621120e-7</td>
<td>8/70/6.505425e-7</td>
<td>4/76/6.414078e-7</td>
<td>74/76/9.621120e-7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>(4, 0, ..., 0)</th>
<th>(20, 0, ..., 20)</th>
<th>(30, 0, ..., 30)</th>
<th>(-4, 0, ..., -4)</th>
<th>(-20, 0, ..., -20)</th>
<th>(-30, 0, ..., -30)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dim</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
</tr>
<tr>
<td>$n = 500$</td>
<td>67/69/7.145027e-7</td>
<td>73/75/7.057076e-7</td>
<td>5/77/6.163480e-7</td>
<td>7/69/7.145024e-7</td>
<td>73/75/7.057075e-7</td>
<td>75/77/6.163479e-7</td>
</tr>
<tr>
<td>$n = 800$</td>
<td>69/71/6.188110e-7</td>
<td>75/77/6.115054e-7</td>
<td>5/77/9.172559e-7</td>
<td>9/71/6.188106e-7</td>
<td>5/77/6.115053e-7</td>
<td>75/77/9.172558e-7</td>
</tr>
</tbody>
</table>

Test Result for the Problem 2

Table 3. Small-scale.

<table>
<thead>
<tr>
<th>$x_i$</th>
<th>(1, ..., 1)</th>
<th>(3, ..., 3)</th>
<th>(4, ..., 4)</th>
<th>(1, 0, ..., 0)</th>
<th>(3, 0, ..., 0)</th>
<th>(4, 0, ..., 0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dim</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
<td>NI/NG/GG</td>
</tr>
<tr>
<td>$n = 10$</td>
<td>20/22/3.007469e-7</td>
<td>38/47/6.088293e-7</td>
<td>44/48/8.898591e-7</td>
<td>0/23/3.452856e-7</td>
<td>5/41/5.833715e-7</td>
<td>29/34/4.338894e-7</td>
</tr>
</tbody>
</table>
results are quite well for the test Problems with the proposed method. The initial points and the dimension don’t influence the performance of the algorithm 1 very much. However, we find the started points will influence the result for the problem 2 a little in our experiment. In one word, the numerical are attractively. The method can be used to the system of nonlinear equations whose Jacobian is not symmetric.

5. Conclusion

In this paper, we propose a new nonmonotone line search method for symmetric nonlinear equations. The global convergence is proved and the numerical results show that this technique is interesting. The reason is that the new nonmonotone line search algorithm used fewer function and gradient evaluations, on average, than either the monotone or the traditional nonmonotone scheme. We hope the method will be a further topic for symmetric nonlinear equations.

6. Acknowledgements

We would like to thank these referees for giving us many valuable suggestions and comments that improve this paper greatly.

7. References


