A Limited BFGS Trust-region Method with a New Nonmonotone Technique for Nonlinear Equations

Meihong Zhou, Qinghua Zhou*
College of Mathematics and Information Science, Hebei University, Baoding, 071002, China

Abstract: In this paper, we incorporate a nonmonotone technique with the new proposed adaptive trust region radius for solving the nonlinear equations. To decrease the computational complexity, a limited memory BFGS update is used to generate an approximated matrix rather than a normal Jacobian matrix or quasi-Newton matrix. Moreover, a line search technique is used to avoid repeatedly computing the trust region algorithm. Theoretical analysis indicates that the new method preserves the global convergence under mild conditions.

Keywords: Nonlinear equations; Trust region method; Nonmonotone strategy; Limited memory BFGS method; Global convergence

1. Introduction

Consider the following nonlinear system of equations:

$$F(x)=0,\ x\in R^n, \tag{1}$$

where $F: R^n \to R^n$ is a continuously differentiable mapping in the form $F(x)=(F_1(x),F_2(x),\ldots,F_n(x))^T$. There are various methods to solve the above nonlinear systems and the trust-region method is a very popular way among them (see [1-5] for instance).

Suppose that $F(x)$ has a zero, then the nonlinear system (1) is equivalent to the following nonlinear unconstrained least-squares problem

$$\min f(x):=\frac{1}{2}\|F(x)\|^2 \quad s.t.\ x\in R^n \tag{2}$$

where $\|.\|$ denotes the Euclidean norm.

At each iterative point x_k, the traditional TR methods obtain the trial step d_k using the following subproblem model:

$$\min q_k(d)=\frac{1}{2}\|F_k+J_kd\|^2 \quad s.t.\ d\in R^n \text{ and } \|d\|\le\Delta_k. \tag{3}$$

where $f_k=f(x_k), F_k=F(x_k), J_k=F'(x_k)$, and $\Delta_k>0$ is trust region radius. Because the Jacobian matrix $F'(x)$ must be computed at all iterations, which obviously increases the workload, we can use the update matrix generated by the quasi-Newton method instead of Jacobian matrix. Yuan et al. [3] raised a new TR subproblem defined by

$$\min_{d\in R^n} q_k(d)=\frac{1}{2}\|F_k+B_kd\|^2 \quad s.t.\ \|d\|\le\Delta_k. \tag{4}$$

where $\Delta_k=c^p\|F_k\|, c\in(0,1), p>0$ is an integer and B_k is generated by the BFGS formula

$$B_{k+1}=B_k-\frac{B_ks_ks_k^TB_k}{s_k^TB_ks_k}+\frac{y_ky_k^T}{y_k^Ts_k}, \tag{5}$$

where $s_k=x_{k+1}-x_k$ and $y_k=F_{k+1}-F_k$. The subproblem (3) can be also rewritten as follows

$$\min_{d\in R^n} q_k(d)=\frac{1}{2}\|F_k+J_kd\|^2=\frac{1}{2}\|F_k\|^2+F_k^TJ_kd+\frac{1}{2}d^TJ_k^TJ_kd \quad s.t.\ \|d\|\le\Delta_k.$$

because the Jacobian matrix $F'(x_k)$ is symmetric. Yuan et al. [6] propose the following TR subproblem model:

$$\min_{d\in R^n} q_k(d)=\frac{1}{2}\|F_k+J_kd\|^2=\frac{1}{2}\|F_k\|^2+F_k^TB_kd+\frac{1}{2}d^TB_k'd \quad s.t.\ \|d\|\le\Delta_k.$$

where B_k' is defined by the special BFGS update:

$$B_{k+1}'=B_k'-\frac{B_k's_ks_k^TB_k'}{s_k^TB_k's_k}+\frac{d_kd_k^T}{d_k^Ts_k}, \tag{6}$$

where $d_k=F(x_k+y_k)-F(x_k)$. These TR methods require the matrix information (the Jacobian matrix or the BFGS update matrix), which will increase the computational complexity, so we will use the limited-memory BFGS (L-M-BFGS) method instead of the BFGS update. The L-M-BFGS update formula is defined as:

$$
\begin{aligned}
N_{k+1} &= V_k^T N_k V_k + r_k s_k s_k^T \\
&= V_k^T \left[V_{k-1}^T N_{k-1} V_{k-1} + r_{k-1} s_{k-1} s_{k-1}^T \right] V_k + r_k s_k s_k^T \\
&= \mathbf{L} \\
&= \left[V_k^T \mathbf{L} V_{k-\tilde{m}+1} \right] N_{k-\tilde{m}+1} \left[V_{k-\tilde{m}+1} \mathbf{L} V_k^T \right] \\
&\quad + r_{k-\tilde{m}+1} \left[V_{k-1}^T \mathbf{L} V_{k-\tilde{m}+2} \right] s_{k-\tilde{m}+1} s_{k-\tilde{m}+1}^T \left[V_{k-\tilde{m}+2} \mathbf{L} V_{k-1}^T \right] \\
&\quad + \mathbf{L} \\
&\quad + r_k s_k s_k^T .
\end{aligned} \tag{7}
$$

where $r_k = \frac{1}{s_k^T y_k}, V_k = 1 - r_k y_k s_k^T, \tilde{m}(>0)$ is an inter, $N_0 = I$ is the unit matrix. The difference between the L-M-BFGS method and the BFGS method is that the inverse Hessian approximation is not explicitly formed, but defined by a small number of BFGS updates. And it provides a fast rate of linear convergence and requires minimal memory. Therefore, the L-M-BFGS TR subproblem model is designed as follows

$$
\begin{aligned}
&\min_{d \in R^n} q_k(d) = \frac{1}{2} \left\| F_k + N_k^{-1} d \right\|^2 \\
&s.t. \quad \|d\| \le \Delta_k .
\end{aligned} \tag{8}
$$

where $\Delta_k = c^p \|F_k\|, c \in (0,1), p > 0$ is an integer.

As we all know, the nonmonotone technique can improve the iterative algorithms in optimization. In this paper, we propose a modified trust-region method for solving nonlinear equations. The method is motivated by the new nonmonotone technique proposed in [7]. From [7], we have

$$
R_k = h_k f_{l(k)} + (1 - h_k) f_k \tag{9}
$$

where

$f_{l(k)} = \max_{0 \le j \le m(k)} \{ f_{k-j} \}, \quad k = 0,1,2,\mathbf{K},$

$m(0) = 0, \ 0 \le m(k) \le \min\{ m(k-1)+1, N \}, N \ge 0$

$h_k \in [h_{\min}, h_{\max}] \ \ for \ \ h_{\min} \in [0,1), \ \ h_{\max} \in [h_{\min}, 1].$

Then, we define the actual reduction as

$ared_k(d_k) = f(x_k + d_k) - R_k$

and the predict reduction as

$pred_k(d_k) = q_k(d_k) - q_k(0)$.

Then, we define the following ratio

$$
r_k = \frac{ared_k(d_k)}{pred_k(d_k)} = \frac{f(x_k + d_k) - R_k}{q_k(d_k) - q_k(0)} \tag{10}
$$

In the traditional TR methods, if $r_k \ge m_1$, the trial step d_k is accepted and it is called as a successful iteration. Otherwise, we need to solve the TR subproblem repeatedly, which increase the computational cost. Therefore, in this paper, similar with [8], we propose an inexact nonmonotone line search technique

$$
f(x_k + a_k d_k) \le R_k + e_k - g a_k^2 f(x_k) \tag{11}
$$

where $g \in (0,1), R_0 = f_0$, the positive sequence $\{e_k\}$ satisfies $\sum_{k=0}^{\infty} e_k < \infty$.

If $r_k < m_1$, the next point is defined by $x_{k+1} = x_k + a_k d_k$, where $a_k = r^{i_k}, r \in (0,1), i_k$ is the smallest non-negative integer i such that (11).

Furthermore, the adaptive radius which control the size of the trust-region radius to prevent increasing and decreasing the radius, can also improve the TR methods. If the trust-region radius Δ_k is very large, then the number of subproblems will be increased and the workload may be increased. On the other hand, if Δ_k is very small, then the total number of iterations is increased and efficiency of the method will be possibly reduced. Keyvan Amini et al. [9] proposed the following adaptive radius:

$$
\Delta_{k+1} := \begin{cases} h_1 a_k \Delta_k^* & if \ r_k < m_1, \\ NF_{l(k+1)} & if \ m_1 \le r_k < m_2, \\ h_2 NF_{l(k+1)} & if \ r_k \ge m_2, \end{cases} \tag{12}
$$

in which

$$
NF_{l(k)} := \max_{0 \le j \le n(k)} \left\{ \|F_{k-j}\| \right\}, \quad k \in N \mathbf{U} \{0\}, \tag{13}
$$

and

$n(0) := 0 \ and \ 0 \le n(k) \le \min\{ n(k-1)+1, N \} \ with \ N > 0$,

$0 < m_1 < m_2 < 1, \ 0 < h_1 < 1 < h_2, \ \Delta_k^* = \max\{ \|F_k\|, \Delta_k \}$.

Because the elements of the new sequence generated by $\{NF_{l(k)}\}_{k \ge 0}$ are always larger than the elements of $\{\|F_k\|\}_{k \ge 0}$, the trust-region radius cannot become too small as possible whenever iterates are not near the optimum. On the other hand, this sequence is decrease and so it prevents the radius of trust-region staying too large whenever iterates are not far away from the optimum.

In this paper, we incorporate a nonmonotone technique with the new proposed adaptive trust region radius. Moreover, we use a limited memory BFGS update to generate an approximated matrix. And under mild conditions, the convergence analysis is established.

The rest of this paper is organized as follows. In Section 2, the new algorithm will be introduced. The convergence analysis is investigated in Section 3. Finally, some conclusions are addressed in Section 4.

2. Algorithm

To obtain better convergence, we will define a new trust region model in which the trust region radius differs from that of the normal method. Motivated by the observations in Section 1, we present the following trust region subproblem that we use to obtain d_k :

$$
\begin{aligned}
&\min_{d \in R^n} q_k(d) = \frac{1}{2} \left\| F_k + N_k^{-1} d \right\|^2 \\
&s.t. \quad \left\| N_k^{-1} d \right\| \le \Delta_k^* .
\end{aligned} \tag{14}
$$

where $\Delta_k^* = \max\{\|F_k\|, \Delta_k\}$.

Now, we outline our algorithm as follows:

Algorithm 1

Initial: Choose a starting point $x_0 \in R^n$, an initial symmetric positive definite matrix $N_0 \in R^{n \times n}$, positive integer m_1, and constants $e > 0, 0 < m_1 < m_2 < 1, 0 < h_1 < 1 < h_2$, $g \in (0,1), J_0 := J(x_0), \Delta_0 := NF_0, NF_0 := \|F_0\|, f_0 = 1/2\|F_0\|^2, F_0 := F(x_0), n(0) := 0, k := 0.$

Step1: If $\|F_k\| < e$ holds, stop; otherwise, let $\Delta_k^* = \max\{\|F_k\|, \Delta_k\}$, and go to step 2.

Step2: Solve trust region subproblem (14) and obtain d_k.

Step3: Compute $m(k), f_{l(k)}, R_k$ by (9) and r_k by (10). If $r_k \geq m_1$, let $x_{k+1} = x_k + d_k$;

Otherwise, let $x_{k+1} = x_k + a_k d_k$, where $a_k = r^{i_k}$ and i_k is the smallest non-negative integer i such that (11) holds for $a = r^i$.

Step4:Let $\tilde{m} = \min\{k+1, m_1\}, s_k = x_{k+1} - x_k = d_k, y_k = F_{k+1} - F_k.$ We update N_0 $\tilde{m}$ times to obtain N_{k+1} by (7).

Step5: $F_{k+1} := F(x_{k+1});\ f_{k+1} := f(x_{k+1});\ J_{k+1} := J(x_{k+1});$ compute $n(k+1)$ and $NF_{l(k+1)}$ according with (13); determine Δ_{k+1} using (12).

Step 6: Set $k := k+1$ and go to step 1.

Algorithm 2

Initial: An initial symmetric positive definite matrix $B_0 \in R^{n \times n}$.

Step4:Let $\tilde{m} = \min\{k+1, m_1\}, s_k = x_{k+1} - x_k = d_k, y_k = F_{k+1} - F_k.$ We update B_0 $\tilde{m}$ times, i.e., for $l = k - \tilde{m} + 1, \mathbf{L}, k$ compute

$$B_k^{l+1} = B_k^l - \frac{B_k^l s_l s_l^T B_k^l}{s_l^T B_k^l s_l} + \frac{y_l y_l^T}{y_l^T s_l}, \qquad (15)$$

where $s_l = x_{l+1} - x_l, y_l = J_{l+1} - J_l$ and $B_k^{k-\tilde{m}+1} = B_0$ for all k.

Note Algorithm 1 and 2 are mathematically equivalent. Throughout this paper, we only discuss the global convergence of Algorithm 2.

3. Convergence Analysis

This section gives some convergence results under the following assumptions.

(H1) Let the level set $\Omega = \{x \mid f(x) \leq f(x_0)\}$ be bounded.

(H2) $F(x)$ is continuously differentiable, $\{\|F_k\|\}$ is bounded, and J_k is uniformly nonsingular.

(H3) $\{B_k\}$ generated by Algorithm 2 is positive definite and bounded, i.e., there exist positive constants $M \geq m > 0$ such that

$$\|B_k d\| \leq M\|d\|,\ m\|d\|^2 \leq d^T B_k d,\ \ d \in R^n. \qquad (16)$$

If $\{B_k\}$ generated by Algorithm 2 is not positive definite, we can utilize various
methods [10,11] to ensure this property.

Lemma 3.1 If d_k is the solution of (14), then

$$-pred_k(d_k) \geq \frac{1}{2}\|B_k F_k\| \min\left\{\Delta_k^*, \frac{\|B_k F_k\|}{\|B_k^T B_k\|}\right\} \qquad (17)$$

holds.

Proof. The proof is similar to Lemma 3.1 in [12].

Considering Algorithm 2, by (17) and (10), for a successful iteration, we obtain

$$f(x_k + d_k) \leq R_k - \frac{1}{2} m_1 \|B_k F_k\| \Delta_k^*. \qquad (18)$$

Lemma 3.2 Suppose that the sequence $\{x_k\}$ is generated by Algorithm 2. Then, we have

$$f_{k+1} \leq R_{k+1} \quad \forall k \in N \mathbf{U} \{0\}. \qquad (19)$$

Proof. The proof is similar to Lemma 3.2 in [7].

REMARK 3.1 Since $e_k > 0$, after a finite number of reductions of a_k, the condition $f(x_k + a_k d_k) \leq f_k + e_k - g a_k^2 f(x_k)$ necessarily holds. From Lemma 3.2 we know that $f_{k+1} \leq R_{k+1} \quad \forall k \in N \mathbf{U} \{0\}$. So the line search process, i.e. Step 3 of Algorithm 1, is well defined. Thus, we have the following reasonable assumption.

(H4) There exists a constant a_* that satisfies

$$a_k \geq a_*, \quad \forall k. \qquad (20)$$

Lemma 3.3 Let (H1)-(H3) hold and $\{x_k\}$ be generated by Algorithm 2. Then $\{x_k\} \subset \Omega$. Moreover, $\{f(x_k)\}$ converges.

Proof. The proof is similar to Lemma 3.3 in [3].

Lemma 3.4 Let (H1)-(H3) hold, then we have $x_k \in \Omega$ and the sequence $\{f_{l(k)}\}$ is not monotonically increasing. Therefore the sequence $\{f_{l(k)}\}$ is convergent.

Proof. Using the definition of R_k and $f_{l(k)}$, we get

$$R_k = h_k f_{l(k)} + (1\text{-}h_k) f_k \leq h_k f_{l(k)} + (1\text{-}h_k) f_{l(k)} = f_{l(k)} \qquad (21)$$

From (19) and (21), we have

$$f_k \leq R_k \leq f_{l(k)}. \qquad (22)$$

It is clear that $R_0 = f_0$. By induction, we show that $x_k \in \Omega$. We assume that $x_i \in \Omega$, for $i = 1, 2, \mathbf{K}, k$. We then prove that $x_{k+1} \in \Omega$. To do this, we consider two cases:

For $r_k \geq m_1$, we get

$$R_k - f(x_k + d_k) \geq m_1(q_k(0) - q_k(d_k)) \geq 0.$$

For $r_k < m_1$, using (18), we get

$$f(x_k + d_k) \leq R_k - \frac{1}{2} m_1 \|B_k F_k\| \Delta_k \leq R_k.$$

These two inequalities along with (22) show that

$$f_{k+1} \le R_k \le f_{l(k)} \le f_0 \qquad \forall k \in N \cup \{0\}. \tag{23}$$

Thus, we have $x_k \in \Omega$.

Now, we prove that the sequence $\{f_{l(k)}\}$ is not monotonically increasing. To do so, we consider the following two cases:

For $k \ge N$, we have $m(k+1) \le m(k)+1$. Thus, from the definition of $f_{l(k+1)}$ and (22), we can write

$$f_{l(k+1)} = \max_{0\le j\le m(k+1)} \{f_{k-j+1}\} \le \max_{0\le j\le m(k)+1} \{f_{k-j+1}\} = \max\{f_{l(k)}, f_{k+1}\} \le f_{l(k)}. \tag{24}$$

For $k < N$, it is obvious that $m(k)=k$. Since any $k,\ f_k \le f_0$, then we have $f_{l(k)} = f_0$.

Both cases show that the sequence $\{f_{l(k)}\}$ is not monotonically increasing.

Moreover, the conclusions above imply that $f_{l(k)}$ is bounded. So $f_{l(k)}$ is convergent.

Lemma 3.5 Let (H2) and (H3) hold. Then, there exist positive constants $m_1 \le M_1, m_2 \le M_2, and\ m_3 \le M_3$ that satisfy the following inequalities:

$$m_1\|d_k\|^2 \le -d_k^T F_k \le M_1\|d_k\|^2, \tag{25}$$

$$m_2\|d_k\|^2 \le -d_k^T B_k F_k \le M_2\|d_k\|^2, \tag{26}$$

And

$$m_3\|F_k\| \le \|d_k\| \le M_3\|F_k\|. \tag{27}$$

Proof. The proof is similar to the proof of Lemma 3.2 in [12].

Lemma 3.6 Let the sequence $\{x_k, F_k, d_k\}$ be generated by Algorithm2, and let (H1)-(H4) hold. Then, we obtain

$$\sum_{k=0}^{\infty}\left(-F_k^T d_k\right) < \infty. \tag{28}$$

In particular, we obtain

$$\lim_{k\to\infty}\left(-F_k^T d_k\right) = 0. \tag{29}$$

Proof. Considering Algorithm 2.2, for a successful iteration, using (18), (25), (27) and the definition of Δ_k^*, we get

$$f(x_{k+1}) - f_{l(k)} \le f(x_{k+1}) - R_k \le -\frac{1}{2}m_1 m\|F_k\|^2 \le \frac{1}{2}m_1 m\frac{-\|d_k\|^2}{M_3} \le \frac{m_1 m}{2M_1M_3}d_k^T F_k. \tag{30}$$

On the other hand, using the line search (11), we have $f(x_k + a_k d_k) - R_k - e_k \le -g{a_k}^2 f(x_k)$ By (H4), we get

$$\begin{aligned} f(x_{k+1}) - f_{l(k)} - e_k &\le f(x_{k+1}) - R_k - e_k \\ &\le -g{a_k}^2 f(x_k) \le -\frac{1}{2}g{a_*}^2\|F_k\|^2 \qquad (31) \\ &\le \frac{1}{2}g{a_*}^2\frac{-\|d_k\|^2}{M_3^2} \le \frac{1}{2}g{a_*}^2\frac{d_k^T F_k}{m_1 M_3^2}. \end{aligned}$$

By lemma 3.3, lemma 3.4 and $\sum_{k=0}^{\infty} e_k < \infty$, combining (30) and (31), it implies that (28) holds. According to (28), it is easy to deduce (29). The proof is complete.

Theorem 3.1 Let $\{x_k\}$ be generated by Algorithm 2.2, let (H1)-(H4) hold. Then, we have

$$\lim_{k\to\infty}\|F_k\| = 0. \tag{32}$$

Proof. By Lemma 3.6, we get $\lim_{k\to\infty}\left(-F_k^T d_k\right) = 0$

Combining the above equation and (25), we have

$$\lim_{k\to\infty}\|d_k\| = 0. \tag{33}$$

This together with (27) gives (32). The proof is complete.

4. Conclusion

In this work, we introduce a new trust-region algorithm for solving the nonlinear equations by combining a non-monotone technique and a limited memory BFGS update. To decrease the computational complexity, we use a line search technique and the new proposed adaptive trust region radius. Under mild conditions, we obtain the global convergence. There are at least three issues that need further improvement: (i) The first issue which should be considered is the numerical experiments, the numerical results can demonstrate the efficiency of the new method. (ii) The second issue is the choice of the parameters in the proposed algorithms, the values used here are not the only choices. Different options can bring different results . (iii) The last important issue is that the proofs of convergence rate need to be completed. All these topics will be the focus of future work.

References

[1] Conn AR, Gould NIM, Toint PL. Trust region method. Philadelphia (PA): Society for Industr- ial and Applied Mathematics, 2000.

[2] Yuan Y. Trust region algorithm for nonlinear equations. Information, 1998;1:7-21.

[3] Yuan G, Lu X, Wei Z. A BFGS trust-region method for nonlinear equations. Computing, 2011; 92:317–333.

[4] Coleman TF, Li Y. An interior trust-region approach for nonlinear minimization subject to bounds. SIAM J. Optim, 1996;6:418–445.

[5] Bellavia S, Macconi M, Orini B.M. An affine scaling trust-region approach to boundconstrai- ned nonlinear systems. Appl. Numer. Math, 2003;44:257–280.

[6] Yuan G, Lu X, Wei Z. BFGS trust-region method for symmetric nonlinear equations. Journal of Computational and Applied Mathematics, 2009;230:44–58.

[7] Wang H, Pu CGD. A BFGS trust-region method with a new nonmonotone technique for nonlinear equations. Optimization, 2015;64 (4):981-992.

[8] Yuan G, Lu X. A new backtracking inexact BFGS method for symmetric nonlinear equations. Comput. Math. Appl, 2008;55:116–129.

[9] Amini K, Shiker MAK, Kimiaei M. A line search trust-region algorithm with nonmonotone adaptive radius for a system of nonlinear equations. 4OR, 2016;14(2):4-16.

[10] Yuan Y. Trust region algorithm for nonlinear equations. Information, 1998;1:7–21.

[11] Schnavel RB, Eskow E. A new modified Cholesky factorization. SIAM J. Sci. Stat. Comput, 1990;11:1136–1158.

[12] Huang L. limited memory technique using trust regions for nonlinear equations. Math. Com- put. Modell, 2015;39(19):5969-5981.