Journal of Theoretical and Applied Information Technology
20th February 2013. Vol. 48 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A LIMITED MEMORY BFGS ALGORITHM WITH SUPER RELAXATION TECHNIQUE FOR NONLINEAR EQUATIONS

[1]GONGLIN YUAN, [2]ZENGXIN WEI, [3]YONG LI, [4]XUPEI ZHAO

[1]Prof., College of Mathematics and Information Science, Guangxi University, Gonglin Yuan

[2]Prof., College of Mathematics and Information Science, Guangxi University, Zengxin Wei

[3] Assoc. Prof., Department of Mathematics, Baise University, Yong Li

[4] Master Student., College of Mathematics and Information Science, Guangxi University, Xupei zhao

E-mail: [1]glyuan@gxu.edu.cn, [2] zxwei@gxu.edu.cn,
[3] liyong9922@sohu.com, [4] xpzhaogxuedu@163.cn

ABSTRACT

In this paper, a trust-region algorithm combining with the limited memory BFGS (L-BFGS) update is proposed for solving nonlinear equations, where the super relaxation technique(SRT) is used. We choose the next iteration point by SRT. The global convergence without the nondegeneracy assumption is obtained under suitable conditions. Numerical results show that this method is very effective for large-scale nonlinear equations problems.

Keywords: *Algorithm; L-BFGS Update; Super Relax; Nonlinear Equations.*

1. INTRODUCTION

Consider the following system of nonlinear equations:

$$g(x) = 0, x \in \Re^n, \qquad (1.1)$$

Where $g : \Re^n \to \Re^n$ is continuously differenti- able. The problem (1.1) has many applications in engineering, such as nonlinear fitting, function approximating and parameter estimating.

There are many algorithms that have been proposed for (1.1), for examples, Gauss- Newton method [1, 8, 12, 14], Levenberg-Marquardt method [5, 20, 31], trust region method [4, 21], and quasi-Newton method [7, 32], etc.. If the Jacobian matrix $\nabla g(x)$ of $g(x)$ is symmetric for all $x \in \Re^n$, then this problem is called symmetric nonlinear equations. Li and Fukushima[9,10] presented Gauss-Newton-based BFGS methods to solve it. These years we made a further study and got some results (see [22, 23, 25, 26, 27, 28, 30]).

Let φ be the norm function defined by $\varphi(x) = \frac{1}{2} \| g(x) \|^2$. Suppose that $g(x)$ has a zero, then the nonlinear equation problem (1.1) is equivalent to the following global optimization problem

$$\min \varphi(x), x \in \Re^n. \qquad (1.2)$$

The trust region method is one of the effective methods for the above problem, where the traditional trust region methods, at each iterative point x_k, the trial step d_k is obtained by solving the following trust-region subproblem

$$\text{Minimize } p_k(d) \text{ such that } \|d\| \le \Delta, \qquad (1.3)$$

where $p_k(d) = \frac{1}{2}\|g(x_k) + \nabla g(x_k)d\|^2$. The trust region methods are globally and superlinearly convergent under the condition that $\nabla g(x^*)$ (x^* is a solution of (1.1)) is nondegenerate, where nondegeneracy means nonsingularity. Nondegeneracy of $\nabla g(x^*)$ seems a too stringent requirement for the purpose of ensuring superlinear convergence. Then Zhang and Wang [31] give a new trust region method:

$$\text{Minimize } \phi_k(d) \text{ such that } c^p \|g(x_k)\|^\gamma, \qquad (1.4)$$

where $\phi_k(d) = \frac{1}{2}\|g(x_k) + \nabla g(x_k)d\|^2, 0 < c < 1, p$ is a non- negative integer, and $0.5 < \gamma < 1$.

The superlinear convergence is obtained under

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

the local error bound condition which is weaker than the nondegeneracy assumption [see [20]]. However, the global convergence also need the nondegeneracy of $\nabla g(x^*)$. Presently, there is no algorithm which has the property that the iterative sequence generated by the algorithm can satisfy $\|g(x_k)\| \to 0$. Without the assumption that $\nabla g(x^*)$ is nondegenerate. In order to overcome this drawback, Yuan et al. [27] presented the trust region subproblem

$$\text{Minimize } h_k(d) \text{ such that } \|d\| \le \Delta_k, \qquad (1.5)$$

Where $h_k(d) = \frac{1}{2}\|g(x_k) + B_k d\|^2$, the radius of trust region Δ_k is defined by $\Delta_k = c^p \|g(x_k)\|, c \in (0, 1)$, p is a nonnegative integer, and B_k is generated by BFGS formula. Comparing with the first two methods, the third method uses the BFGS update matrix instead of the Jacobian matrix. It is well known that the trust-region method will be very useful with the situation when the exact Jacobian or Hessian computation is inexpensive or possible. However, this case is very inferquent in many practices. Then the third trust region method is more efficient than the first two methods for normal practical problems. However, all of these three methods need to store and compute the matrix (the Jacobian or the BFGS update matrix) at every iterations in the algorithm. It can clearly increase the storage and workload in computation, especially for large-scale problems. One of the effective methods to overcome this insufficient is reducing the matrix term. In this paper, we give a trust region method based on the method of [27] which does not compute and store the matrix completely.

Limited memory quasi-Newton methods are known to be effective techniques for solving certain classes of large-scale unconstrained optimization problems (see Buckley and Le Nir [3], Liu and Nocedal [11], Gilbert and Lemar´echal [6], and Byrd, Nocedal, and Schnabel[2]). They make simple approximations of Hessian matrices, which are often good enough to provide a fast rate of linear convergence, and require minimal storage. The implementation is almost identical to that of the standard BFGS method, the only difference is that the inverse Hessian approximation is not formed explicitly, but defined by a small number of BFGS updates. Thus it often provides a fast rate of convergence and requires minimal storage. Some authors have made a study about this technique (see [15, 18, 19, 24, 28, 29] etc.).

Inspired by the above observations, we present a new method, which not only possesses the global convergence without the nondegeneracy assumption under suitable conditions, but also does not compute the matrix completely at every iterations. Naturally, we use limited memory quasi-Newton update matrix instead of the normal matrix (the Jacobian or the quasi-Newton matrix). The given algorithm has the following attributes.

•the matrix of the trust region subproblem is generated by limited memory quasi-Newton update.

•the the next iteration point is determined by super relaxation technique.

•the global convergence without the nondegeneracy assumption under suitable conditions is established.

• numerical results show that the given algorithm is very effective for largescale problems.

Here and throughout this paper, we use the following notations. $\|.\|$ Denote the Euclidian norm of vectors or its induced matrix norm, and $g(x_k)$ is replaced by g_k.

This paper is organized as follows. In Section 2, the algorithm is represented. In Section 3, we prove some convergent results. The numerical results of the method are reported in Section 4.

2. MOTIVATIONS AND ALGORITHM

In this section, we will introduce the limited memory BFGS (L-BFGS) method and the super relaxation technique (SRT) respectively. Then we give the proposed algorithm.

2.1 The limited memory BFGS method

The L-BFGS method is an adaptation of the BFGS method to large-scale problems. The implementation described by Liu and Nocedal [11] is almost identical to that of the standard BFGS method-the only difference is in the matrix update, for getting Hessian inverse approximate H_{k+1}, instead of storing matrices H^k, at every iteration x_k the method stores a small number, say m, of correction pairs $\{s_i, y_i\}$, $i = k-1, \text{K}, k-m$, where $s_k = x_{k+1} - x_k, y_k = g^{k+1} - g^k$.

Where g^k and g^{k+1} denote the gradients of the objective function $f(x)$ at x_k and x_{k+1}, respectively. We consider here only BFGS since we have considerable computational experience in the unconstr-

Journal of Theoretical and Applied Information Technology
20th February 2013. Vol. 48 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ained case indicating the limited memory BFGS performs well. Let $\rho_k = \frac{1}{y_k^T s_k}$ and $U_k = I - \rho_k y_k s_k^T$. If we use the stored correction pairs, then

$$\begin{aligned} & H_{k+1} \\ & = U_k^T \left[(U_{K-1})^T H_{k-1} U_{k-1} + \rho_{k-1} s_{k-1} s_{k-1}^T \right] U_k + \rho_k s_k s_k^T \\ & = V_k^T U_{k-1}^T H_{k-1} U_{k-1} + U_k^T \rho_{k-1} s_{k-1} s_{k-1}^T U_k + \rho_k s_k s_k^T \\ & = \mathrm{L} \\ & = [U_k^T \ \mathrm{L} \ U_{k-m+1}^T] H_{k-m+1} [U_{k-m+1} \ \mathrm{L} \ U_k] + \rho_{k+m-1} [U_{k-1}^T \\ & \mathrm{L} \ U_{k-m+2}^T] s_{k-m+1} s_{k-m+1}^T [U_{k-m+2} \ \mathrm{L} \ U_{k-1}] + \mathrm{L} + \rho_k s_k s_k^T \end{aligned} \quad (2.1)$$

Where $s_k = x_{k+1} - x_k$ and $y_k = g^\alpha(x_{k+1}, \varepsilon_{k+1}) - g^\alpha(x_k, \varepsilon_k)$ To maintain the positive definiteness of the L-BFGS matrix, some researches discard a correction pair $[s_k, y_k]$ if the curvature $s_k^T y_k > 0$ is not satisfied (see [18]). Another approach was proposed by Powell [17] where y_k is defined by

$$y_k = \begin{cases} y_k, & if \ s_k^T y_k \geq 0.2 s_k^T B_k s_k \\ \theta_k y_k + (1-\theta_k) B_k s_k, & others, \end{cases} \quad (2.2)$$

Where $\theta_k = \frac{0.8 s_k^T B_k s_k}{s_k^T B_k s_k - s_k^T y_k}$ and $B_k = (H_k)^{-1}$.

2.2 The Super Relaxation Technique

The super relaxation technique (SRT) is often used in computation mathematics to improve the accuracy, where two target values are generally chosen as a weighted average. The heart of the super relaxation technique is the development of advantage and the inhibition of inferior in the relaxation process, where the relaxation factor plays an important role. However, this technique is rarely used in numerical optimization methods. In the algorithm of this paper, we will use the SRT to improve the efficiency for large scale nonlinear equations.

Based on the above discussions, the given L-BFGS trust region subproblem is defined by

$$\text{Minimize } q_k(d) \text{ such that } \|d\| \leq \Delta_k, \quad (2.3)$$

Where $q_k(d) = \frac{1}{2}\|g(x_k) + B_k d\|^2$, the radius of trust region Δ_k is defined by $\Delta_k = c^p \|g(x_k)\|$, $c \in (0,1)$, p is a nonnegative integer, and $B_k = H_k^{-1}$ is generated by the L-BFGS formula (2.1) and B_0 is an initial symmetric positive definite matrix. Let d_k^p be the solution of (2.3) corresponding to p, and let x_k be the *kth* iteration point. Then the next point is defined by the SRT. (1) Updated point: $x_{k+1}^d = x_k + d_k^p$. (2) Super relaxation point:

$x_{k+1} = w x_k + (1-w) x_{k+1}^d, w \in [0,1]$ is the super relaxation factor.

According to the new trust region subproblem (2.3) and the above super relaxation point, we define the actual reduction as

$$Ared_k(d_k^p) = \varphi(x_k + d_k^p) - \varphi(x_k), \quad (2.4)$$

and the predict reduction as

$$\Pr ed_k(d_k^p) = q_k(d_k^p) - q_k(0). \quad (2.5)$$

Now we the algorithm for solving (1.1) is given as follows.

• Algorithm 1.(L-BFGS trust region with SRT)

Initial: Given constants $\rho, c \in [0,1]$ $w \in [0,1]$, $p = 0, \varepsilon > 0, x_0 \in \Re^n$, $B_0 = H_0^{-1} = I \in \Re^n \times \Re^n$ is symmetric and positive definite. Let k := 0;

Step 1: If $\|g_k\| < \varepsilon$, stop. Otherwise, go to step 2;

Step 2: Solve the trust region subproblem (2.3) with $\Delta = \Delta_k$ to get d_k;

Step 3: Calculate $Ared_k(d_k^p)$,and the radio of actual reduction over predict reduction as

$$r_k^p = \frac{Ared_k(d_k^p)}{\Pr ed_k(d_k^p)}. \quad (2.6)$$

If $r_k^p < \rho$, then we let p = p + 1, go to step 2. Otherwise, go to step 4;

Step 4: Let $x_{k+1}^d = x_k + d_k^p$, $x_{k+1} = w x_k + (1-w) x_{k+1}^d$, $y_k = g_{k+1} - g_k$, update $H_{k+1} = B_{k+1}^{-1}$ by (2.1).

Step 5: Set k := k + 1 and p = 0. Go to step 1.

Remark. (i) In this algorithm, the procedure of "Step 2-Step 3-Step 2" is named as inner cycle.

(ii) In order to ensure the update matrix H_k generated by (2.1), the technique (2.2) of [17] is used.

(iii) In Step 4, if $w = 0$, then $x_{k+1} = x_{k+1}^d = x_k + d_k^p$, the given algorithm is the normal L-BFGS trust region algorithm.

3. CONVERGENCE ANALYSIS

This section will analyze some convergence results. Similar to Mor´e [13], Zhang and Wang [31], or Yuan et al.[27], the following result holds. Here

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

we also give the detail proof.

Lemma 3.1 If d_k^p is the solution of (2.3), then

$$-Pred_k(d_k^p) \geq \frac{1}{2}\|B_k g_k\| \min\{\Delta_k, \frac{\|B_k g_k\|}{\|B_k\|^2}\}. \quad (3.1)$$

Proof. Since d_k^p is the solution of (2.3), for any $\alpha \in [0,1]$, we get

$$\begin{aligned} -Pred_k(d_k^p) &\geq -Pred_k(-\alpha \frac{\Delta_k}{\|B_k g_k\|} B_k g_k) \\ &= \alpha\Delta_k \|B_k g_k\| - \frac{1}{2}\alpha^2\Delta_k^2 (B_k B_k g_k)^T \\ &\quad (B_k B_k g_k)/\|B_k g_k\|^2 \\ &\geq \alpha\Delta_k \|B_k g_k\| - \frac{1}{2}\alpha^2\Delta_k^2 \|B_k B_k\|. \end{aligned}$$

Thus, we have

$$\begin{aligned} -Pred_k(d_k^p) &\geq \max_{0\leq\alpha\leq 1}[\alpha\Delta_k \|B_k g_k\| - \frac{1}{2}\alpha^2\Delta_k^2 \|B_k\|^2] \\ &\geq \frac{1}{2}\|B_k g_k\| \min\{\Delta_k, \frac{\|B_k g_k\|}{\|B_k\|^2}\}. \end{aligned}$$

The proof is complete.

In order to get the global convergence, similar to [27, 31], the following assumption conditions are needed.

Assumption i (A) Let the level set Ω

$$\Omega = \{x \,|\, \varphi(x) \leq \varphi(x_0)\} \quad (3.2)$$

be bounded.

(B) $g(x)$ is twice continuously differentiable on an open convex set Ω_1 containing Ω. The normal function $\varphi(x) = \frac{1}{2}\| g(x_k)\|^2$ is descent along the direction d_k^p, *ie.*,

$$\nabla\varphi(x_k)^T d_k^p \leq 0. \quad (3.3)$$

(C) The following relation

$$\|[\nabla g(x_k) - B_k] g_k\| = O(\|d_k^p\|) \quad (3.4)$$

Holds.

(D) The matrices $\{B_k\}$ are uniformly bounded on Ω_1 ,which means that there exist positive constants $0 < M_0 \leq M$ and $0 < m_1$ such that

$$M_0 \leq \|B_k\| \leq M, \|B_k^{-1}\| \leq m_1 \ \forall k. \quad (3.5)$$

Assumption i(B) implies that there exists $M_1 > 0$ such that

$$\|\nabla g(x_k)^T \nabla g(x_k)\| \leq M_1, \forall k. \quad (3.6)$$

Similar to [27], we have the following two lemmas, here we only state them but omit the proof.

Lemma 3.2 (Lemma 3.1 of [27]) If d_k^p is the solution of (2.3). Let Assumption i hold and $\{x_k\}$ be generatedby Algorithm 1. Then

$$|Ared_k(d_k^p) - Pred_k(d_k^p)| = O(\|d_k^p\|^2).$$

Lemma 3.3 (Lemma 3.2 of [27]) Let Assumption i hold and $\{x_k\}$ be generated by Algorithm 1. Then Algorithm1 does not circle in the inner cycle infinitely.

Lemma 3.3 shows that the proposed algorithm is well-defined.

Lemma 3.4 Let Assumption i hold and $\{x_k\}$ be generated by Algorithm 1. Then $\{x_k\} \subset \Omega$. Moreover, $\{\varphi(x_k)\}$ converges.

Proof. By Taylor formula and (3.3), we have

$$\begin{aligned} \varphi(x_{k+1}) - \varphi(x_k) &= \varphi[wx_k + (1-w)(x_k + d_k^p)] - \varphi(x_k) \\ &= (1-w)\nabla\varphi(x_k)^T d_k^p + o(\|d_k^p\|) \leq 0 \end{aligned} \quad (3.7)$$

The above relation implies that $\varphi(x_{k+1}) \leq \varphi(x_k) \leq \cdots \leq \varphi(x_0)$. Then we conclude that $\{\varphi(x_k)\}$ converges. The proof is complete.

Similar to [27], we have the global convergence, here state it and give the proof process as follows. The following theorem says that the iterative sequence $\{x_k\}$ generated by Algorithm 1 satisfies $\| g(x_k)\| \to 0$ without the assumption that $\nabla g(x^*)$ is nondegenerate, where x^* is a cluster point of $\{x_k\}$.

Theorem 3.1 Let Assumption i holds, $\{x_k\}$ be generated by the Algorithm 1. Then the algorithm either stops finitely or generates an infinite sequence $\{x_k\}$ such that

$$\lim_{k\to\infty} \| g_k \| = 0. \quad (3.8)$$

Proof. If the algorithm does not stop finitely, suppose that the relation

$$\lim_{k\to\infty} \| B_k g_k \| = 0 \quad (3.9)$$

holds. By (3.5), we get

Journal of Theoretical and Applied Information Technology
20th February 2013. Vol. 48 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$0 \le \| g_k \| = \| B_k^{-1} B_k g_k \| \le \| B_k^{-1} \| \| B_k g_k \| \le m_1 \| B_k g_k \|$, then we can get (3.8). Thus, in order to complete this lemma, we only to obtain (3.9).Assume, on the contrary, that there exists a constant $\varepsilon > 0$ and an subsequence $\{k_j\}$ such that

$$\| B_{k_j} g_{k_j} \| \ge \varepsilon \tag{3.10}$$

Define the index set $K = \{k \mid \| B_k g_k \| \ge \varepsilon\}$. Meanwhile, by Assumption i and $\| B_k g_k \| \ge \varepsilon, (k \in K), \| g_k \|$ $(k \in K)$ is bounded away from 0. Without loss of generality, we can assume $\| g_k \| \ge \varepsilon, \forall k \in K$.

By Lemma 3.1 and the definition of Algorithm 1, we have

$$\sum_{k \in K} [\varphi(x_k) - \varphi(x_k + d_k^p)] \ge -\sum_{k \in K} \rho . \Pr ed_k (\frac{1}{2} d_k^{p_k})$$
$$\ge \sum_{k \in K} \rho . \min\{c^{p_k} \varepsilon, \frac{\varepsilon}{M^2}\} . \varepsilon,$$

where p_k is the largest p value obtained in the inner circle at the iterative point x_k.

Lemma 3.4 shows that $\{\varphi(x_k)\}$ is convergent, then the following relation

$$\sum_{k \in K} \rho . \frac{1}{2} \min\{c^{p_k} \varepsilon, \frac{\varepsilon}{M^2}\} . \varepsilon < +\infty$$

holds. Thus, $p_k \to +\infty$ as $k \to +\infty$ and $k \in K$. Then we assume that $p_k \ge 1$ for all $k \in K$.

Using the determination of $p_k (k \in K)$ in the inner circle, the solution d_k' corresponding to the following subproblem

$$\min \ q_k(d) = \frac{1}{2} \| g(x_k) + B_k d \|^2$$
$$s.t. \qquad \| d \| \le c^{p_k - 1} \| g_k \|, \tag{3.11}$$

is unacceptable. Let $x_{k+1}' = x_k + d_k'$, we get

$$\frac{\varphi(x_k) - \varphi(x_{k+1}')}{-Pred_k(d_k')} < \rho. \tag{3.12}$$

From Lemma 3.1, we have

$$-Pred_k(d_k' \ge \frac{1}{2} \min\{c^{p_k - 1} \varepsilon, \frac{\varepsilon}{M^2}\} . \varepsilon.$$

By Lemma 3.2, we obtain

$$\varphi(x_{k+1}') - \varphi(x_k) - Pred_k(d_k')$$
$$= O(\| d_k' \|^2) = O(c^{2(p_k - 1)}).$$

Therefore,

$$| \frac{\varphi(x_{k+1}') - \varphi(x_k)}{Pred_k(d_k')} - 1 | \le \frac{O(c^{2(p_k - 1)})}{0.5 \min\{c^{p_k - 1} \varepsilon, \frac{\varepsilon}{M^2}\} . \varepsilon}.$$

Since $p_k \to +\infty$ as $k \to +\infty$ and $k \in K$, we have $\frac{\varphi(x_k) - \varphi(x_{k+1}')}{-Pred_k(d_k')} \to 1, k \in K$, this contradicts (3.12). This completes the proof.

4. NUMERICAL RESULTS

In this section, we report results of some large scale numerical experiments with the proposed method. We list the test functions as follows [16]

$$g(x) = (f_1(x), f_2(x), \cdots f_n(x))^T,$$

where these functions have the associated initial guess x_0. Where A of Function 8 is the n×n tridiagonal matrix given by

$$A = \begin{bmatrix} 8 & -1 & & & & \\ -1 & 8 & -1 & & & \\ & -1 & 8 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & \ddots & \ddots & -1 \\ & & & & -1 & 8 \end{bmatrix}$$

In the experiments, the parameters were chosen as $\rho = 0.0001$, $\varepsilon = 10^{-5}$, $c = 0.1$, $\gamma = 0.7$, $w = 0.2$, B_0 is the unit matrix. We obtain d_k by (2.3) from Dogleg method. In the inner circle of Algorithm 1, we will accept the trial step if $p > 5$ holds.We also stop the program if the iteration number is larger than fifteen hundred. The program was coded in MATLAB 7.0. The columns of the tables have the following meaning: Dim: the dimension of the problem. NG: the number of the norm function evaluations. NI: the total number of iterations.

$\|g(x^*)\|$: the final function value at the last point.

From Table 2, we can see that our algorithm performs quite well from these problems, and the dimension does not influence the performance of the presented method obviously. For Problem 3 and Problem 7, we can see that the iteration number and the normal function number of dimension 3000 is less than those of dimension 1000.The reason maybe lie in that the number is limited（ $p \le 5$ ） in the inner cycle.

5. CONCLUSION

In this paper, a L-BFGS trust-region algorithm is presented for large scale nonlinear equations. The

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

global convergence is established. The main work of this paper is extent the super relaxation to the nonlinear equation problems, moreover, the L-BFGS update is used to replace the matrix of the trust-region subproblem.

Table 1 (Test Functions)

1.Trigonometric function	$f_i(x)=2(n+i(1-\cos x_i)-\sin x_i$ $-\sum_{j=1}^{n}\cos x_j)(2\sin x_i-\cos x_i), i=1,2,3,\cdots,n$	$x_0=(\frac{101}{100n},\frac{101}{100n},\cdots,\frac{101}{100n})^T.$
2.Logarithmic function	$f_i(x)=\ln(x_i+1)-\frac{x_i}{n}, i=1,2,3,\cdots,n$	$x_0=(1,1,\cdots,1)^T.$
3.Broyden Tridiagonal function	$f_1(x)=(3-0.5x_1)x_1-2x_2+1,$ $f_i(x)=(3-0.5x_i)x_i-x_{i-1}+2x_{i+1}+1, i=2,3,\cdots,n-1,$ $f_n(x)=(3-0.5x_n)x_n-x_{n-1}+1.$	$x_0=(-1,-1,\cdots,-1)^T.$
4.Trigexp function	$f_1(x)=3x_1^3-2x_2-5+\sin(x_1-x_2)\sin(x_1+x_2),$ $f_i(x)=-x_{i-1}e^{x_{i-1}-x_i}-x_i(4+3x_i^2)+2x_{i+1}$ $+\sin(x_i-x_{i+1})\sin(x_i+x_{i+1})-8, i=2,3,\cdots,n-1,$ $f_n(x)=-x_{n-1}e^{x_{n-1}-x_n}+4x_n-3.$	$x_0=(0,0,\cdots,0)^T.$
5.Strictly convex function	$f_i(x)=e^{x_i}-1, i=1,2,3,\cdots,n$	$x_0=(\frac{1}{n},\frac{2}{n},\cdots,)^T.$
6.Variable dimensioned function	$f_i(x)=x_i-1, i=1,2,3,\cdots,n-2,$ $f_{n-1}(x)=\sum_{j=1}^{n-2}j(x_j-1),$ $f_n(x)=(\sum_{j=1}^{n-2}j(x_j-1))^2.$	$x_0=(1-\frac{1}{n},1-\frac{2}{n},\cdots,)^T.$
7.Discrete boundry value function	$f_1(x)=2x_1+0.5h^2(x_1+h)^3-x_2,$ $f_i(x)=2x_i+0.5h^2(x_i+hi)^3-x_{i-1}+x_{i+1}, i=2,3,\cdots,n-1,$ $f_n(x)=2x_n+0.5h^2(x_n+hn)^3-x_{n-1}, h=\frac{1}{n+1}.$	$x_0=(h(h-1),\cdots,h(nh-1))$
8.Discretized two-point boundry value function	$g(x)=Ax+\frac{1}{(n+1)^2}F(x)$ $F(x)=(F_1(x),F_2(x),\cdots,F_n(x))^T, F_i(x)=\sin x_i-1,$	$x_0=(50,0,50,0,\cdots).$

Table 2 (Test results for Algorithm 1)

Functions	Dim	N1 / NG/$\|\|g(x^*)\|\|$	Functions	Dim	N1 / NG/$\|\|g(x^*)\|\|$
Function 1	$n=500$	9 / 15 / 1.237232e−006	Function 1	$n=500$	9 / 15 / 1.237232e−006
	$n=1000$	9 / 15 / 4.092946e−006		$n=1000$	9 / 15 / 4.092946e−006
	$n=3000$	9 / 15 / 4.183233e−006		$n=3000$	9 / 15 / 4.183233e−006
Function 2	$n=500$	6 / 7 / 9.059705e−007	Function 2	$n=500$	6 / 7 / 9.059705e−007
	$n=1000$	6 / 7 / 1.239486e−006		$n=1000$	6 / 7 / 1.239486e−006
	$n=3000$	6 / 7 / 2.099938e−006		$n=3000$	6 / 7 / 2.099938e−006
Function 3	$n=500$	114 / 120 / 8.063389e−006	Function 3	$n=500$	114 / 120 / 8.063389e−006
	$n=1000$	120 / 126 / 8.418914e−006		$n=1000$	120 / 126 / 8.418914e−006
	$n=3000$	119 / 125 / 8.922338e−006		$n=3000$	119 / 125 / 8.922338e−006
Function 4	$n=500$	60 / 77 / 1.501565e−006	Function 4	$n=500$	60 / 77 / 1.501565e−006
	$n=1000$	58 / 80 / 2.167228e−006		$n=1000$	58 / 80 / 2.167228e−006
	$n=3000$	73 / 85 / 4.746286e−006		$n=3000$	73 / 85 / 4.746286e−006

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

(i) Since the L-BFGS has a fast rate of convergence and requires minimal storage. Then the matrix of the trust region subproblem is generated by L-BFGS update.

(ii) In order to get better numerical results, the SRT is used in the algorithm where the next iteration point is determined by super relaxation technique.

(iii) Similar to \cite{yuan113}, the global convergence without the nondegeneracy assumption under suitable conditions is established.

(iv) In order to show the performance of the given algorithm, large-scale problems are tested. Practical results show that the given algorithm is effective.

ACKNOWLEDGMENT.

This work is supported by Guangxi NSF (Grant No. 2012GXNSFAA053002) and China NSF (Grant No. 11261006, 11161003 and 71001015).

REFERENCES

[1] D. P. Bertsekas, Nonlinear programming, Athena Scientific, Belmont, 1995.

[2] R. H. Byrd, J. Nocedal, and R. B. Schnabel, Representations of quasi-Newton matrices and their use in limited memory methods, Mathematical Programming,63(1994), pp. 129-156.

[3] A. Buchley and A. LeNir, QN-like variable storage conjugate gradients, Mathematical Programming, 27(1983), pp. 577-593.

[4] A. R. Conn, N. I. M. Gould, and P. L.Toint,Trust region method, Society for Industrial and Applied Mathema-tics, Philadelphia PA, 2000.

[5] J. Y. Fan, A modified Levenberg-Marquardt algorithm for singular system of nonlinear equations, Journal of Computational Mathematics, 21(2003), pp. 625-636.

[6] J. C. Gilbert and C. Lemar´echal, Some numerical experiments with variable storage quasi-Newton algorithms, Mathematical Programming, 45(1989), pp. 407-436.

[7] A. Griewank, The 'global' convergence of Broyden-like methods with a suitable line search, Journal of the Australian Mathematical Society, Series B, 28(1986), pp. 75-92.

[8] K. Levenberg, A method for the solution of certain nonlinear problem in least squares. Quarterly of Applied Mathematics, 2(1944), pp. 164-166.

[9] D. Li and M. Fukushima, A global and super-linear convergent Gauss-Newton-based BFGS method for symmetric nonlinear equations, SIAM Journal on Numerical Analysis, 37(1999), 152-172.

[10] D. Li, L. Qi, and S. Zhou, Descent directions of quasi-Newton methods for symmetric nonlinear equations,SIAM Journal on Numerical Analysis, (5)40(2002), pp. 1763-1774.

[11] D. C. Liu and J. Nocedal, On the limited memory BFGS method for large scale optimization, Mathematical Programming, 45(1989), pp. 503-528.

[12] D. W. Marquardt, An algorithm for leastsquares estimation of nonlinear inequalities. SIAM Journal on Applied Mathematics, 11(1963), pp. 431-441.

[13] J. J. Mor´e, Recent development in algorithm and software for trust region methods, in: A. Bachem, M. Grotschel, B. Kortz (Eds.), Mathematical Programming: The State of the Art, Spinger-Verlag, Berlin, 1983, 258-285.

[14] J. Nocedal and S. J. Wright, Numerical optimization, Spinger, New York, 1999.

[15] Q. Ni and Y. Yuan, A subspace limited memory quasi-Newton algorithm for large-scale nonlinear bound consrtained optimization, Mathematics of Computation,66(1997), pp. 1509-1520.

[16] J. M. Ortega and W. C. Rheinboldt, Iterative solution of nonlinear equations in several variables, Academic Press, New York, 1970.

[17] M. J. D. Powell, A fast algorithm for nonlinearly constrained optimization calculations, Numerical Analyis, (1978), pp. 155-157.

[18] R. H. Ryrd, P. H. Lu, J. Nocedal, and C. Y. Zhu, A limited memory algorithm for bound constrained optimization, SIAM Journal on Scientific Computing, 16(1995), pp. 1190-1208.

[19] Y. Xiao and D. Li, An active set limited memory BFGS algorithm for large-scale bound constrained optimization, Mathematical Methods of Operations Research, 67(2008), pp. 443-454.

[20] N. Yamashita and M. Fukushima, On the rate of convergence of the Levenberg-Marquardt Method, Computing, 15(2001), pp. 239-249.

[21] Y. Yuan, Trust region algorithm for nonlinear equations, Information, 1(1998), pp. 7-21.

[22] G. Yuan, A new method with descent property for symmetric nonlinear equations, Numerical functional Analysis and Optimization, 31 (2010) , pp. 974-987.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[23] G. Yuan and X. Lu, A new backtracking inexact BFGS method for symmetric nonlinear equations, Computers and Mathematics with Applications, 55(2008) pp. 116-129.

[24] G. Yuan and X. Lu, An active set limited memory BFGS algorithm for bound constrained optimization,Applied Mathematical Modelling, 35(2011), pp. 3561-3573.

[25] G. Yuan, X. Lu, and Z. Wei, BFGS trust-region method for symmetric nonlinear equations, Journal of Computational and Applied Mathematics, (1)230(2009), pp. 44-58.

[26] G. Yuan, S. Lu, and Z. Wei, A new trust-region method with line search for solving symmetric nonlinear equations, International Journal of Computer Mathematics, 88(2011), pp. 2109-2123.

[27] G. Yuan, Z. Wei, and X. Lu, A BFGS trust-region method for nonlinear equations, Computing, 92(2011), pp. 317-333.

[28] G. Yuan, Z. Wei, and S. Lu, Limited memory BFGS method with backtracking for symmetric nonlinear equations, Mathematical and Computer Modelling, 54(2011), pp. 367-377

[29] G. Yuan, Z. Wei, and Y. Wu, Modified limited memory BFGS method with nonmonotone line search for unconstrained optimization, Journal of the Korean Mathematical Society, 47(2010), pp. 767-788.

[30] G. Yuan and S. Yao, A BFGS algorithm for solving symmetric nonlinear equations, Optimization,DOI:10.1080/02331934.2011.564621.

[31] J. Zhang and Y. Wang, A new trust region method for nonlinear equations, Mathematical Methods of Operations Research, 58(2003), pp. 283-298.

[32] D. Zhu, Nonmonotone backtracking inexact quasi-Newton algorithms for solving smmooth nonlinear equations, Applied Mathematics and Computation,161(2005),pp.875-895.