Journal of Inequalities and Applications
a SpringerOpen Journal

RESEARCH Open Access

A tensor trust-region model for nonlinear system

Songhua Wang[1*] and Shulun Liu[2]

*Correspondence:
WSH6600789@126.com
[1]School of Mathematics and Statistics, Baise University, Baise, P.R. China
Full list of author information is available at the end of the article

Abstract

It has turned out that the tensor expansion model has better approximation to the objective function than models of the normal second Taylor expansion. This paper conducts a study of the tensor model for nonlinear equations and it includes the following: (i) a three dimensional symmetric tensor trust-region subproblem model of the nonlinear equations is presented; (ii) the three dimensional symmetric tensor is replaced by interpolating function and gradient values from the most recent past iterate, which avoids the storage of the three dimensional symmetric tensor and decreases the workload of the computer; (iii) the limited BFGS quasi-Newton update is used instead of the second Jacobian matrix, which generates an inexpensive computation of a complex system; (iv) the global convergence is proved under suitable conditions. Numerical experiments are done to show that this proposed algorithm is competitive with the normal algorithm.

MSC: 65K05; 90C26

Keywords: Tensor model; Trust region; Nonlinear equations; BFGS formula; Convergence

1 Introduction

This paper focuses on

$$S(x) = 0, \quad x \in \Re^n, \tag{1.1}$$

where $S : \Re^n \to \Re^n$ is continuously differentiable nonlinear system. The nonlinear system (1.1) has been proved to possess wildly different application fields in parameter estimating, function approximating, and nonlinear fitting, etc. At present, there exist many effective algorithms working in it, such as the traditional Gauss–Newton method [1, 9–11, 14, 16], the BFGS method [8, 23, 27, 29, 39, 43], the Levenberg–Marquardt method [6, 24, 42], the trust-region method [4, 26, 35, 41], the conjugate gradient algorithm [12, 25, 30, 38, 40], and the limited BFGS method [13, 28]. Here and in the next statement, for research convenience, suppose that $S(x)$ has solution x^*. Setting $\beta(x) := \frac{1}{2}\|S(x)\|^2$ as a norm function, the problem (1.1) is equivalent to the following optimization problem:

$$\min \beta(x), \quad x \in \Re^n. \tag{1.2}$$

The trust-region (TR) methods have as a main objective solving the so-called trust-region subproblem model to get the trial step d_k,

$$\begin{aligned} &\text{Min} \quad Tp_k(d) = \frac{1}{2}\left\|S(x_k) + \nabla S(x_k)d\right\|^2, \\ &\|d\| \leq \triangle, \end{aligned}$$

where x_k is the kth iteration, $\triangle$ is the so-called TR radius, and $\|\cdot\|$ is the normally Euclidean norm of vectors or matrix. The first choice for many scholars is to study the above model to make a good improvement. An adaptive TR model is designed by Zhang and Wang [42]:

$$\begin{aligned} &\text{Min} \quad \phi_k(d) = \frac{1}{2}\left\|S(x_k) + \nabla S(x_k)d\right\|^2, \\ &\|d\| \leq c^p\left\|S(x_k)\right\|^\gamma, \end{aligned}$$

where $p > 0$ is an integer, and $0 < c < 1$ and $0.5 < \gamma < 1$ are constants. Its superlinear convergence is obtained under the local error bound assumption, by which it has been proved that the local error bound assumption is weaker than the nondegeneracy [24]. Thus one made progress in theory. However, its global convergence still needs the nondegeneracy. Another adaptive TR subproblem is defined by Yuan et al. [35]:

$$\begin{aligned} &\text{Min} \quad Tq_k(d) = \frac{1}{2}\left\|S(x_k) + B_k d\right\|^2, \\ &\|d\| \leq c^p\left\|S(x_k)\right\|, \end{aligned} \tag{1.3}$$

where B_k is generated by the BFGS quasi-Newton formula

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k {y_k}^T}{{y_k}^T s_k}, \tag{1.4}$$

where $y_k = S(x_{k+1}) - S(x_k)$, $s_k = x_{k+1} - x_k$, x_{k+1} is the next iteration, and B_0 is an initial symmetric positive definite matrix. This TR method can possess the global convergence without the nondegeneracy, which shows that this paper made a further progress in theory. Furthermore, it also possesses the quadratic convergence. It has been showed that the BFGS quasi-Newton update is very effective for optimization problems (see [32, 33, 36] etc.). There exist many applications of the TR methods (see [19–21, 31] etc.) for nonsmooth optimizations and other problems.

It is not difficult to see that the above models only get the second Taylor expansion and approximation. Can we get the approximation to reach one more level, namely the third expansion, or even the fourth? The answer is positive and a third Taylor expansion is used and a three dimensional symmetric tensor model is stated. In the next section, the motivation and the tensor TR model are stated. The algorithm and its global convergence are presented in Sect. 3. In Sect. 4, we do the experiments of the algorithms. One conclusion is given in the last section.

2 Motivation and the tensor trust-region model

Consider the tensor model for the nonlinear system $S(x)$ at x_k,

$$\vartheta(x_k + d) = S(x_k) + \nabla S(x_k)^T d + \frac{1}{2} T_k d^2, \tag{2.1}$$

where $\nabla S(x_k)$ is the Jacobian matrix of $S(x)$ at x_k and T_k is three dimensional symmetric tensor. It is not difficult to see that the above tensor model (2.1) has better approximation than the normal quadratical trust-region model. It has been proved that the tensor is significantly simpler when only information from one past iterate is used (see [3] for details), which obviously decreases the complexity of the computation of the three dimensional symmetric tensor T_k. Then the model (2.1) can be written as the following extension:

$$\vartheta(x_k + d) = S(x_k) + \nabla S(x_k)^T d + \frac{3}{2}\left(s_{k-1}^T d\right)^2 s_{k-1}. \tag{2.2}$$

In order to avoid the exact Jacobian matrix $\nabla S(x_k)$, we use the quasi-Newton update matrix B_k instead of it. Thus, our trust-region subproblem model is designed by

$$\begin{aligned} &\text{Min} \quad N_k(d) = \frac{1}{2}\left\| S(x_k) + B_k d + \frac{3}{2}\left(s_{k-1}^T d\right)^2 s_{k-1} \right\|^2, \\ &\|d\| \le c^p \left\| S(x_k) \right\|^\gamma, \end{aligned} \tag{2.3}$$

where $B_k = H_k^{-1}$ and H_k is generated by the following low-storage limited BFGS (L-BFGS) update formula:

$$\begin{aligned} H_{k+1} &= V_k^T H_k V_k + \rho_k s_k s_k^T \\ &= V_k^T \left[V_{k-1}^T H_{k-1} V_{k-1} + \rho_{k-1} s_{k-1} s_{k-1}^T \right] V_k + \rho_k s_k s_k^T \\ &= \cdots \\ &= \left[V_k^T \cdots V_{k-m+1}^T \right] H_{k-m+1} [V_{k-m+1} \cdots V_k] \\ &\quad + \rho_{k-m+1} \left[V_{k-1}^T \cdots V_{k-m+2}^T \right] s_{k-m+1} s_{k-m+1}^T [V_{k-m+2} \cdots V_{k-1}] \\ &\quad + \cdots \\ &\quad + \rho_k s_k s_k^T, \end{aligned} \tag{2.4}$$

where $\rho_k = \frac{1}{s_k^T y_k}$, $V_k = I - \rho_k y_k s_k^T$, I is the unit matrix and m is a positive integer. It has turned out that the L-BFGS method has a fast linear convergence rate and minimal storage, and it is effective for large-scale problems (see [2, 13, 28, 34, 37] etc.). Let d_k^p be the solution of (2.3) corresponding to the constant p. Define the actual reduction by

$$Ad_k\left(d_k^p\right) = \beta\left(x_k + d_k^p\right) - \beta(x_k), \tag{2.5}$$

and the predict reduction by

$$Pd_k\left(d_k^p\right) = N_k\left(d_k^p\right) - N_k(0). \tag{2.6}$$

Based on definition of the actual reduction $Ad_k(d_k^p)$ and the predict reduction $Pd_k(d_k^p)$, their radio is defined by

$$r_k^p = \frac{Ad_k(d_k^p)}{Pd_k(d_k^p)}. \tag{2.7}$$

Therefore, the tensor trust-region model algorithm for solve (1.1) is stated as follows.

Algorithm 1

Initial: Constants $\rho, c \in (0,1)$, $p = 0$, $\epsilon > 0$, $x_0 \in \Re^n$, $m > 0$, and $B_0 = H_0^{-1} \in \Re^n \times \Re^n$ is a symmetric and positive definite matrix. Let $k := 0$;

Step 1: Stop if $\|S(x_k)\| < \epsilon$ holds;

Step 2: Solve (2.3) with $\triangle = \triangle_k$ to obtain d_k^p;

Step 3: Compute $Ad_k(d_k^p)$, $Pd_k(d_k^p)$, and the radio r_k^p. If $r_k^p < \rho$, let $p = p + 1$, go to Step 2. If $r_k^p \geq \rho$, go to the next step;

Step 4: Set $x_{k+1} = x_k + d_k^p$, $y_k = S(x_{k+1}) - S(x_k)$, update $B_{k+1} = H_{k+1}^{-1}$ by (2.4) if $y_k^T d_k^p > 0$, otherwise set $B_{k+1} = B_k$;

Step 5: Let $k := k + 1$ and $p = 0$. Go to Step 1.

Remark The procedure of "Step 2–Step 3–Step 2" is called the inner cycle in the above algorithm. It is necessary for us to prove that the inner cycle is finite, which generates the circumstance that Algorithm 1 is well defined.

3 Convergence results

This section focuses on convergence results of Algorithm 1 under the following assumptions.

Assumption i

(A) The level set Ω defined by

$$\Omega = \{x \mid \beta(x) \leq \beta(x_0)\} \tag{3.1}$$

is bounded.

(B) On an open convex set Ω_1 containing Ω, the nonlinear system $S(x)$ is twice continuously differentiable.

(C) The approximation relation

$$\left\|\left[\nabla S(x_k) - B_k\right] S(x_k)\right\| = O\left(\left\|d_k^p\right\|\right) \tag{3.2}$$

is true, where d_k^p is the solution of the model (2.3).

(D) On Ω_1, the sequence matrices $\{B_k\}$ are uniformly bounded, namely there exist constants $0 < M_0 \leq M$ satisfying

$$M_s \leq \|B_k\| \leq M_l \quad \forall k. \tag{3.3}$$

Assumption i (B) means that there exists a constant $M_L > 0$ satisfying

$$\left\|\nabla S(x_k)^T \nabla S(x_k)\right\| \leq M_L, \quad \forall k. \tag{3.4}$$

Based on the above assumptions and the definition of the model (2.3), we have the following lemma.

Lemma 3.1 *Let d_k^p be the solution of* (2.3), *then the inequality*

$$Pd_k\left(d_k^p\right) \le -\frac{1}{2}\left\|B_k S(x_k)\right\| \min\left\{\triangle_k, \frac{\|B_k S(x_k)\|}{M_l^2}\right\} + O\left(\triangle_k^2\right) \tag{3.5}$$

holds.

Proof By the definition of d_k^p of (2.3), then, for any $\alpha \in [0,1]$, we get

$$\begin{aligned}
Pd_k\left(d_k^p\right) &\le Pd_k\left(-\alpha \frac{\triangle_k}{\|B_k S(x_k)\|} B_k S(x_k)\right) \\
&= \frac{1}{2}\left[\alpha^2 \triangle_k^2 \frac{\|B_k B_k S(x_k)\|^2}{\|B_k S(x_k)\|^2} + \alpha^4 \triangle_k^4 \frac{9}{4} \frac{(s_{k-1}^T B_k S(x_k))^4}{\|B_k S(x_k)\|^4}\right. \\
&\quad + 3\alpha^2 \triangle_k^2 \frac{(s_{k-1}^T B_k S(x_k))^2}{\|B_k s_{k-1}\|^2} S(x_k)^T s_{k-1} - 2\alpha \triangle_k \frac{(S(x_k)^T B_k B_k S(x_k))}{\|B_k S(x_k)\|} \\
&\quad \left. - 3\alpha^3 \triangle_k^3 \frac{(s_{k-1}^T B_k S(x_k))^2 s_{k-1}^T B_k B_k S(x_k)}{\|B_k S(x_k)\|^3}\right] \\
&= \frac{1}{2}\left[\alpha^2 \triangle_k^2 \frac{\|B_k B_k S(x_k)\|^2}{\|B_k S(x_k)\|^2} - 2\alpha \triangle_k \frac{(S(x_k)^T B_k B_k S(x_k))}{\|B_k S(x_k)\|} + O\left(\triangle_k^2\right)\right] \\
&\le -\alpha \triangle_k \left\|B_k S(x_k)\right\| + \frac{1}{2}\alpha^2 \triangle_k^2 M_l^2 + O\left(\triangle_k^2\right).
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
Pd_k\left(d_k^p\right) &\le \min_{0\le\alpha\le 1}\left[-\alpha \triangle_k \left\|B_k S(x_k)\right\| + \frac{1}{2}\alpha^2 \triangle_k^2 M_l^2\right] + O\left(\triangle_k^2\right) \\
&\le -\frac{1}{2}\left\|B_k S(x_k)\right\| \min\left\{\triangle_k, \frac{\|B_k S(x_k)\|}{M_l^2}\right\} + O\left(\triangle_k^2\right).
\end{aligned}$$

The proof is complete. □

Lemma 3.2 *Let d_k^p be the solution of* (2.3). *Suppose that Assumption* i *holds and $\{x_k\}$ is generated by Algorithm* 1. *Then we have*

$$\left|Ad_k\left(d_k^p\right) - Pd_k\left(d_k^p\right)\right| = O\left(\left\|d_k^p\right\|^2\right).$$

Proof Using Assumption i, the definition of (2.5) and (2.6), we obtain

$$\begin{aligned}
&\left|Ad_k\left(d_k^p\right) - Pd_k\left(d_k^p\right)\right| \\
&= \left|\beta\left(x_k + d_k^p\right) - N_k\left(d_k^p\right)\right| \\
&= \frac{1}{2}\left|\left\|S(x_k) + \nabla S(x_k) d_k^p + O\left(\left\|d_k^p\right\|^2\right)\right\|^2 - \left\|S(x_k) + B_k d_k^p + \frac{3}{2}\left(s_{k-1}^T d_k^p\right)^2 s_{k-1}\right\|^2\right| \\
&= \left|S(x_k)^T \nabla S(x_k) d_k^p - S(x_k)^T B_k d_k^p + O\left(\left\|d_k^p\right\|^2\right) + O\left(\left\|d_k^p\right\|^3\right) + O\left(\left\|d_k^p\right\|^4\right)\right|
\end{aligned}$$

$$\leq \left\| \left[\nabla S(x_k) - B_k \right] S(x_k) \right\| \left\| d_k^p \right\| + O\left(\left\| d_k^p \right\|^2 \right) + O\left(\left\| d_k^p \right\|^3 \right) + O\left(\left\| d_k^p \right\|^4 \right)$$
$$= O\left(\left\| d_k^p \right\|^2 \right).$$

This completes the proof. □

Lemma 3.3 *Let the conditions of Lemma 3.2 hold. We conclude that Algorithm 1 does not infinitely circle in the inner cycle* (*"Step 2–Step 3–Step 2"*).

Proof This lemma will be proved by contradiction. Suppose, at x_k, that Algorithm 1 infinitely circles in the inner cycle, namely, $r_k^p < \rho$ and $c^p \to 0$ with $p \to \infty$. This implies that $\|g_k\| \geq \epsilon$, or the algorithm stops. Thus we conclude that $\|d_k^p\| \leq \triangle_k = c^p \|g_k\| \to 0$ is true.

By Lemma 3.1 and Lemma 3.2, we get

$$\left| r_k^p - 1 \right| = \frac{|Ad_k(d_k^p) - Pd_k(d_k^p)|}{|Pd_k(d_k^p)|}$$
$$\leq \frac{2O(\|d_k^p\|^2)}{\triangle_k \|B_k S(x_k)\| + O(\triangle_k^2)} \to 0.$$

Therefore, for p sufficiently large, we have

$$r_k^p \geq \rho, \tag{3.6}$$

which generates a contradiction with the fact $r_k^p < \rho$. The proof is complete. □

Lemma 3.4 *Suppose that the conditions of Lemma 3.3 holds. Then we conclude that* $\{x_k\} \subset \Omega$ *is true and* $\{\beta(x_k)\}$ *converges.*

Proof By the results of the above lemma, we get

$$r_k^p \geq \rho > 0. \tag{3.7}$$

Combining with Lemma 3.1 generates

$$\beta(x_{k+1}) \leq \beta(x_k) \leq \cdots \leq \beta(x_0).$$

Then $\{x_k\} \subset \Omega$ holds. By the case $\beta(x_k) \geq 0$, we deduce that $\{\beta(x_k)\}$ converges. This completes its proof. □

Theorem 3.5 *Suppose that the conditions of Lemma 3.3 hold and* $\{x_k\}$ *is generated by Algorithm 1. Then Algorithm 1 either finitely stops or generates an infinite sequence* $\{x_k\}$ *satisfying*

$$\lim_{k \to \infty} \left\| S(x_k) \right\| = 0. \tag{3.8}$$

Proof Suppose that Algorithm 1 does not finitely stop. We need to obtain (3.8). Assume that

$$\lim_{k \to \infty} \left\| B_k S(x_k) \right\| = 0 \tag{3.9}$$

holds. Using (3.3) one gets (3.8). So, we can complete this lemma by (3.9). We use the contradiction to have (3.9). Namely, we suppose that there exist an subsequence $\{k_j\}$ and a positive constant ε such that

$$\left\|B_{k_j}S(x_{k_j})\right\| \geq \varepsilon. \tag{3.10}$$

Let $K = \{k \mid \|B_kS(x_k)\| \geq \varepsilon\}$ be an index set. Using Assumption i, the case $\|B_kS(x_k)\| \geq \varepsilon$ $(k \in K)$, and $\|S(x_k)\|$ $(k \in K)$ is bounded away from 0, we assume

$$\left\|S(x_k)\right\| \geq \varepsilon, \quad \forall k \in K$$

holds. By Lemma 3.1 and the definition of Algorithm 1, we obtain

$$\begin{aligned}\sum_{k\in K}\left[\beta(x_k) - \beta(x_{k+1})\right] &\geq -\sum_{k\in K}\rho Pd_k\left(d_k^{p_k}\right)\\ &\geq \sum_{k\in K}\rho\frac{1}{2}\min\left\{c^{p_k}\varepsilon, \frac{\varepsilon}{M_l^2}\right\}\varepsilon,\end{aligned}$$

where p_k is the largest p value obtained in the inner circle. Lemma 3.4 tells us that the sequence $\{\beta(x_k)\}$ is convergent, thus

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

Then $p_k \to +\infty$ when $k \to +\infty$ and $k \in K$. Therefore, for all $k \in K$, it is reasonable for us to assume $p_k \geq 1$. In the inner circle, by the determination of p_k $(k \in K)$, let d'_k corresponding to the subproblem

$$\begin{aligned}&\text{Min} \quad q_k(d) = \frac{1}{2}\left\|S(x_k) + B_kd + \frac{3}{2}\left(s_{k-1}^Td\right)^2s_{k-1}\right\|^2,\\ &\text{s.t.} \quad \|d\| \leq c^{p_k-1}\left\|S(x_k)\right\|,\end{aligned} \tag{3.11}$$

be unacceptable. Setting $x'_{k+1} = x_k + d'_k$ one has

$$\frac{\beta(x_k) - \beta(x'_{k+1})}{-Pd_k(d'_k)} < \rho. \tag{3.12}$$

Using Lemma 3.1 and the definition $\triangle_k$ one has

$$-Pd_k\left(d'_k\right) \geq \frac{1}{2}\min\left\{c^{p_k-1}\varepsilon, \frac{\varepsilon}{M_l^2}\right\}\varepsilon.$$

Using Lemma 3.2 one gets

$$\beta\left(x'_{k+1}\right) - \beta(x_k) - Pd_k\left(d'_k\right) = O\left(\left\|d'_k\right\|^2\right) = O\left(c^{2(p_k-1)}\right).$$

Thus, we obtain

$$\left|\frac{\beta(x'_{k+1})-\beta(x_k)}{Pd_k(d'_k)}-1\right| \le \frac{O(c^{2(p_k-1)})}{0.5\min\{c^{p_k-1}\varepsilon, \frac{\varepsilon}{M_l^2}\}\varepsilon + O(c^{2(p_k-1)}\varepsilon^2)}.$$

Using $p_k \to +\infty$ when $k \to +\infty$ and $k \in K$, we get

$$\frac{\beta(x_k)-\beta(x'_{k+1})}{-Pd_k(d'_k)} \to 1, \quad k \in K,$$

this generates a contradiction to (3.12). This completes the proof. □

4 Numerical results

This section reports some numerical results of Algorithm 1 and the algorithm of [35] (Algorithm YL).

4.1 Problems

The nonlinear system obeys the following statement:

$$S(x) = \big(g_1(x), g_2(x), \ldots, g_n(x)\big)^T.$$

Problem 1 Trigonometric function

$$g_i(x) = 2\left(n + i(1-\cos x_i) - \sin x_i - \sum_{j=1}^{n}\cos x_j\right)(2\sin x_i - \cos x_i), \quad i = 1,2,3,\ldots,n.$$

Initial guess: $x_0 = (\frac{101}{100n}, \frac{101}{100n}, \ldots, \frac{101}{100n})^T$.

Problem 2 Logarithmic function

$$g_i(x) = \ln(x_i+1) - \frac{x_i}{n}, \quad i = 1,2,3,\ldots,n.$$

Initial points: $x_0 = (1,1,\ldots,1)^T$.

Problem 3 Broyden tridiagonal function ([7], pp. 471–472)

$$\begin{aligned} g_1(x) &= (3-0.5x_1)x_1 - 2x_2 + 1,\\ g_i(x) &= (3-0.5x_i)x_i - x_{i-1} + 2x_{i+1} + 1, \quad i = 2,3,\ldots,n-1,\\ g_n(x) &= (3-0.5x_n)x_n - x_{n-1} + 1. \end{aligned}$$

Initial points: $x_0 = (-1,-1,\ldots,-1)^T$.

Problem 4 Trigexp function ([7], p. 473)

$$\begin{aligned} g_1(x) &= 3x_1^3 + 2x_2 - 5 + \sin(x_1 - x_2)\sin(x_1 + x_2),\\ g_i(x) &= -x_{i-1}e^{x_{i-1}-x_i} + x_i\big(4 + 3x_i^2\big) + 2x_{i+1} \end{aligned}$$

$$+ \sin(x_i - x_{i+1}) \sin(x_i + x_{i+1}) - 8, \quad i = 2, 3, \ldots, n-1,$$

$$g_n(x) = -x_{n-1} e^{x_{n-1} - x_n} + 4x_n - 3.$$

Initial guess: $x_0 = (0, 0, \ldots, 0)^T$.

Problem 5 Strictly convex function 1 ([18], p. 29). $S(x)$ is the gradient of $h(x) = \sum_{i=1}^{n} (e^{x_i} - x_i)$. We have

$$g_i(x) = e^{x_i} - 1, \quad i = 1, 2, 3, \ldots, n.$$

Initial points: $x_0 = (\frac{1}{n}, \frac{2}{n}, \ldots, 1)^T$.

Problem 6 Strictly convex function 2 ([18], p. 30). $S(x)$ is the gradient of $h(x) = \sum_{i=1}^{n} \frac{i}{10} \times (e^{x_i} - x_i)$. We have

$$g_i(x) = \frac{i}{10} \left(e^{x_i} - 1\right), \quad i = 1, 2, 3, \ldots, n.$$

Initial guess: $x_0 = (1, 1, \ldots, 1)^T$.

Problem 7 Penalty function

$$g_i(x) = \sqrt{10^{-5}} (x_i - 1), \quad i = 1, 2, 3, \ldots, n-1,$$

$$g_n(x) = \left(\frac{1}{4n}\right) \sum_{j=1}^{n} x_j^2 - \frac{1}{4}.$$

Initial guess: $x_0 = (\frac{1}{3}, \frac{1}{3}, \ldots, \frac{1}{3})^T$.

Problem 8 Variable dimensioned function

$$g_i(x) = x_i - 1, \quad i = 1, 2, 3, \ldots, n-2,$$

$$g_{n-1}(x) = \sum_{j=1}^{n-2} j(x_j - 1),$$

$$g_n(x) = \left(\sum_{j=1}^{n-2} j(x_j - 1)\right)^2.$$

Initial guess: $x_0 = (1 - \frac{1}{n}, 1 - \frac{2}{n}, \ldots, 0)^T$.

Problem 9 Discrete boundary value problem [15]

$$g_1(x) = 2x_1 + 0.5h^2 (x_1 + h)^3 - x_2,$$

$$g_i(x) = 2x_i + 0.5h^2 (x_i + hi)^3 - x_{i-1} + x_{i+1}, \quad i = 2, 3, \ldots, n-1,$$

$$g_n(x) = 2x_n + 0.5h^2 (x_n + hn)^3 - x_{n-1},$$

$$h = \frac{1}{n+1}.$$

Initial points: $x_0 = (h(h-1), h(2h-1), \ldots, h(nh-1))$.

Problem 10 The discretized two-point boundary value problem similar to the problem in [17]

$$S(x) = Ax + \frac{1}{(n+1)^2}F(x) = 0,$$

with A is the $n \times n$ tridiagonal matrix given by

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

and $F(x) = (F_1(x), F_2(x), \ldots, F_n(x))^T$ with $F_i(x) = \sin x_i - 1$, $i = 1, 2, \ldots, n$, and $x = (50, 0, 50, 0, \ldots)$.

Parameters: $\rho = 0.05$, $\epsilon = 10^{-4}$, $c = 0.5$, $p = 3$, $m = 6$, H_0 is the unit matrix.

The method for (1.3) and (2.3): the *Dogleg* method [22].

Codes experiments: run on a PC with an Intel Pentium(R) Xeon(R) E5507 CPU @2.27 GHz, 6.00 GB of RAM, and the Windows 7 operating system.

Codes software: MATLAB r2017a.

Stop rules: the program stops if $\|S(x)\| \leq 1e{-}4$ holds.

Other cases: we will stop the program if the iteration number is larger than a thousand.

4.2 Results and discussion

The column meaning of the tables is as follows.

Dim: the dimension.

NI: the iterations number.

NG: the norm function number.

Time: the CPU-time in s.

Numerical results of Table 1 show the performance of these two algorithms as regards NI, NG and Time. It is not difficult to see that:

(i) Both of these algorithms can successfully solve all these ten nonlinear problems;

(ii) the NI and the NG of these two algorithm do not increase when the dimension becomes large;

(iii) the NI and the NG of Algorithm 1 are competitive to those of Algorithm YL and the Time of Algorithm YL is better than that of Algorithm 1. To directly show their the efficiency, the tool of [5] is used and three figures for NI, NG and Time are listed.

Figures 1–3 show the performance of NI, NG and Time of these two algorithms. It is easy to see that the NI and the NG of Algortihm 1 have won since their performance profile plot is on top right. And the Time of Algorithm YL has superiority to Algorithm 1. Both of these two algorithms have good robustness. All these three figures show that both of these two algorithms are very interesting and we hope they will be further studied in the future.

Table 1 Experiment results

Nr	Dim	Algorithm 1			Algorithm YL		
		Ni	NG	Time	NI	NG	Time
1	400	9	18	10.93567	11	22	1.778411
	800	9	18	52.46314	11	22	7.176046
	1600	8	14	215.453	11	22	42.57267
2	400	4	10	11.27887	6	7	1.185608
	800	4	10	45.94229	6	7	4.071626
	1600	4	10	251.38	6	7	22.58894
3	400	4	10	2.808018	64	125	8.642455
	800	4	10	10.74847	78	129	52.26034
	1600	4	10	70.80885	68	99	262.5653
4	400	2	2	0.8112052	6	17	1.092007
	800	2	2	2.839218	6	22	3.08882
	1600	2	2	14.08689	6	22	13.27569
5	400	3	6	1.731611	6	7	0.936006
	800	3	6	5.616036	6	7	3.650423
	1600	3	6	30.32659	6	7	22.44854
6	400	3	6	1.279208	5	6	0.7176046
	800	3	6	5.397635	5	16	2.88601
	1600	3	6	29.88979	5	16	16.39571
7	400	5	14	3.790824	12	49	1.435209
	800	5	14	22.52654	12	49	4.69563
	1600	5	14	102.0403	17	83	19.23492
8	400	1	2	1.294808	3	6	0.2808018
	800	1	2	5.694037	3	6	0.8580055
	1600	1	2	31.091	3	6	3.775224
9	400	13	19	11.01367	12	15	1.60681
	800	9	15	40.95026	11	17	7.191646
	1600	10	19	299.3191	10	16	38.07984
10	400	3	9	2.558416	40	50	12.44888
	800	3	9	11.62207	40	50	49.43672
	1600	3	9	73.07087	41	53	365.7911

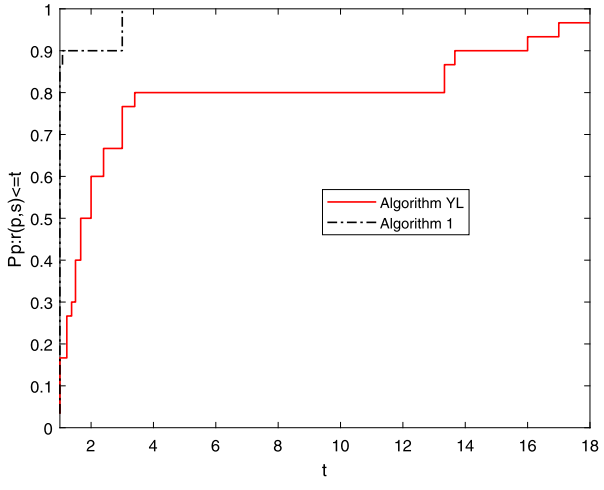

Figure 1 Performance profiles of these methods (NI)

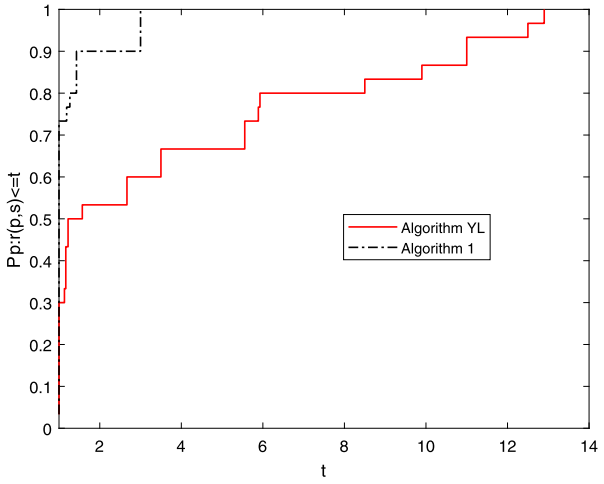

Figure 2 Performance profiles of these methods (NG)

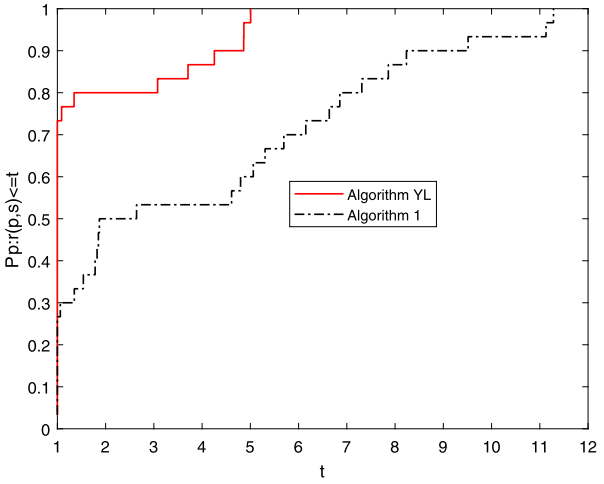

Figure 3 Performance profiles of these methods (Time)

5 Conclusions

This paper considers the tensor trust-region model for nonlinear system. The global convergence is obtained under suitable conditions and numerical experiments are reported. This paper includes the following main work:

(1) a tensor trust-region model is established and discussed.

(2) the low workload update is used in this tensor trust-region model. In the future, we think this tensor trust-region model shall be more significant.

Acknowledgements

The authors would like to thank the above the support funding. The authors also thank the referees and the editor for their valuable suggestions which greatly improve our paper.

Funding

This work was supported by the National Natural Science Fund of China (Grant No. 11661009).

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
Dr. Songhua Wang mainly analyzed the theory results and Shulun Liu has done the numerical experiments. All authors read and approved the final manuscript.

Authors' information
Songhua Wang and Shulun Liu are co-first authors.

Author details
[1]School of Mathematics and Statistics, Baise University, Baise, P.R. China. [2]Department of Information Engineering, Jiyuan Vocational and Technical College, Henan, P.R. China.

Publisher's Note

Received: 17 October 2018 Accepted: 3 December 2018 Published online: 13 December 2018

References

1. Bertsekas, D.P.: Nonlinear Programming. Athena Scientific, Belmont (1995)
2. Byrd, R.H., Nocedal, J., Schnabel, R.B.: Representations of quasi-Newton matrices and their use in limited memory methods. Math. Program. **63**, 129–156 (1994)
3. Chow, T., Eskow, E., Schnabel, R.: Algorithm 783: a software package for unstrained optimization using tensor methods. ACM Trans. Math. Softw. **20**, 518–530 (1994)
4. Conn, A.R., Gould, N.I.M., Toint, P.L.: Trust Region Method. SIAM, Philadelphia (2000)
5. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)
6. Fan, J.Y.: A modified Levenberg–Marquardt algorithm for singular system of nonlinear equations. J. Comput. Math. **21**, 625–636 (2003)
7. Gomez-Ruggiero, M., Martinez, J.M., Moretti, A.: Comparing algorithms for solving sparse nonlinear systems of equations. SIAM J. Sci. Stat. Comput. **23**, 459–483 (1992)
8. Griewank, A.: The 'global' convergence of Broyden-like methods with a suitable line search. J. Aust. Math. Soc. Ser. B **28**, 75–92 (1986)
9. Levenberg, K.: A method for the solution of certain nonlinear problem in least squares. Q. Appl. Math. **2**, 164–166 (1944)
10. Li, D., Fukushima, M.: A global and superlinear convergent Gauss–Newton-based BFGS method for symmetric nonlinear equations. SIAM J. Numer. Anal. **37**, 152–172 (1999)
11. Li, D., Qi, L., Zhou, S.: Descent directions of quasi-Newton methods for symmetric nonlinear equations. SIAM J. Numer. Anal. **40**(5), 1763–1774 (2002)
12. Li, X., Wang, S., Jin, Z., Pham, H.: A conjugate gradient algorithm under Yuan–Wei–Lu line search technique for large-scale minimization optimization models. Math. Probl. Eng. **2018**, Article ID 4729318 (2018)
13. Li, Y., Yuan, G., Wei, Z.: A limited-memory BFGS algorithm based on a trust-region quadratic model for large-scale nonlinear equations. PLoS ONE **10**(5), Article ID e0120993 (2015)
14. Marquardt, D.W.: An algorithm for least-squares estimation of nonlinear inequalities. SIAM J. Appl. Math. **11**, 431–441 (1963)
15. Moré, J.J., Garbow, B.S., Hillström, K.E.: Testing unconstrained optimization software. ACM Trans. Math. Softw. **7**, 17–41 (1981)
16. Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, New York (1999)
17. Ortega, J.M., Rheinboldt, W.C.: Iterative Solution of Nonlinear Equations in Several Variables. Academic Press, New York (1970)
18. Raydan, M.: The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. SIAM J. Optim. **7**, 26–33 (1997)
19. Sheng, Z., Yuan, G.: An effective adaptive trust region algorithm for nonsmooth minimization. Comput. Optim. Appl. **71**, 251–271 (2018)
20. Sheng, Z., Yuan, G., et al.: An adaptive trust region algorithm for large-residual nonsmooth least squares problems. J. Ind. Manag. Optim. **14**, 707–718 (2018)
21. Sheng, Z., Yuan, G., et al.: A new adaptive trust region algorithm for optimization problems. Acta Math. Sci. **38B**(2), 479–496 (2018)
22. Wang, Y.J., Xiu, N.H.: Theory and Algoithm for Nonlinear Programming. Shanxi Science and Technology Press, Xian (2004)
23. Wei, Z., Yuan, G., Lian, Z.: An approximate Gauss–Newton-based BFGS method for solving symmetric nonlinear equations. Guangxi Sciences **11**(2), 91–99 (2004)
24. Yamashita, N., Fukushima, M.: On the rate of convergence of the Levenberg–Marquardt method. Computing **15**, 239–249 (2001)
25. Yuan, G., Duan, X., Liu, W., Wang, X., Cui, Z., Sheng, Z.: Two new PRP conjugate gradient algorithms for minimization optimization models. PLoS ONE **10**(10), Article ID e0140071 (2015). https://doi.org/10.1371/journal.pone.0140071
26. Yuan, G., Lu, S., Wei, Z.: A new trust-region method with line search for solving symmetric nonlinear equations. Int. J. Comput. Math. **88**, 2109–2123 (2011)
27. Yuan, G., Lu, X.: A new backtracking inexact BFGS method for symmetric nonlinear equations. Comput. Math. Appl. **55**, 116–129 (2008)

28. Yuan, G., Lu, X.: An active set limited memory BFGS algorithm for bound constrained optimization. Appl. Math. Model. **35**, 3561–3573 (2011)
29. Yuan, G., Lu, X., Wei, Z.: BFGS trust-region method for symmetric nonlinear equations. J. Comput. Appl. Math. **230**(1), 44–58 (2009)
30. Yuan, G., Meng, Z., Li, Y.: A modified Hestenes and Stiefel conjugate gradient algorithm for large-scale nonsmooth minimizations and nonlinear equations. J. Optim. Theory Appl. **168**, 129–152 (2016)
31. Yuan, G., Sheng, Z.: Nonsmooth Optimization Algorithms. Press of Science, Beijing (2017)
32. Yuan, G., Sheng, Z., Wang, P., Hu, W., Li, C.: The global convergence of a modified BFGS method for nonconvex functions. J. Comput. Appl. Math. **327**, 274–294 (2018)
33. Yuan, G., Wei, Z.: Convergence analysis of a modified BFGS method on convex minimizations. Comput. Optim. Appl. **47**, 237–255 (2010)
34. Yuan, G., Wei, Z., Lu, S.: Limited memory BFGS method with backtracking for symmetric nonlinear equations. Math. Comput. Model. **54**, 367–377 (2011)
35. Yuan, G., Wei, Z., Lu, X.: A BFGS trust-region method for nonlinear equations. Computing **92**, 317–333 (2011)
36. Yuan, G., Wei, Z., Lu, X.: Global convergence of the BFGS method and the PRP method for general functions under a modified weak Wolfe–Powell line search. Appl. Math. Model. **47**, 811–825 (2017)
37. Yuan, G., Wei, Z., Wu, Y.: Modified limited memory BFGS method with nonmonotone line search for unconstrained optimization. J. Korean Math. Soc. **47**, 767–788 (2010)
38. Yuan, G., Wei, Z., Yang, Y.: The global convergence of the Polak–Ribiere–Polyak conjugate gradient algorithm under inexact line search for nonconvex functions. J. Comput. Appl. Math. (2018). https://doi.org/10.1016/j.cam.2018.10.057
39. Yuan, G., Yao, S.: A BFGS algorithm for solving symmetric nonlinear equations. Optimization **62**, 45–64 (2013)
40. Yuan, G., Zhang, M.: A three-terms Polak–Ribière–Polyak conjugate gradient algorithm for large-scale nonlinear equations. J. Comput. Appl. Math. **286**, 186–195 (2015)
41. Yuan, Y.: Trust region algorithm for nonlinear equations. Information **1**, 7–21 (1998)
42. Zhang, J., Wang, Y.: A new trust region method for nonlinear equations. Math. Methods Oper. Res. **58**, 283–298 (2003)
43. Zhu, D.: Nonmonotone backtracking inexact quasi-Newton algorithms for solving smooth nonlinear equations. Appl. Math. Comput. **161**, 875–895 (2005)