Scholars Journal of Engineering and Technology (SJET) **ISSN 2321-435X (Online)**
Sch. J. Eng. Tech., 2015; 3(5A):529-534 **ISSN 2347-9523 (Print)**
©Scholars Academic and Scientific Publisher
(An International Publisher for Academic and Scientific Resources)
www.saspublisher.com

Research Article

A New Non-monotone Self-Adaptive Trust Region Method with Fixed Step-size for Unconstrained Optimization

Changyuan Li, Qinghua Zhou*
College of Mathematics and Information Science, Hebei University, Baoding, Hebei Province, China

***Corresponding author**
Qinghua Zhou
Email: qinghua.zhou@gmail.com

Abstract: In this paper, we propose and analyze a new non-monotone self-adaptive trust region method with fixed step-size for unconstrained optimization. Unlike the traditional non-monotone trust region method, our algorithm utilizes a fixed formula to get the next iterative point if a trial step is not adopted. Besides, the trust region radius of related sub-problem adjusts itself adaptively. By the above techniques, we can decrease the number of solving sub-problems efficiently. Under some standard assumptions, we show that the new proposed method has a global convergence.
Keywords: unconstrained optimization, non-monotone technique, self-adaptive trust region method, fixed step-size, global convergence

1. Introduction

Consider the following unconstrained optimization problem:

$$\min f(x), \quad x \in R^n, \tag{1}$$

where $f : R^n \to R$ is a twice continuously differentiable function. Throughout this paper, we use the following notation:

- $\|\cdot\|$ is the Euclidean norm.
- $g(x) = \nabla f(x) \in R^n$ and $H(x) \in R^{n\times n}$ are the gradient and Hessian matrix of f evaluated at x, respectively.
- $f_k = f(x_k), g_k = g(x_k), H_k = \nabla^2 f(x_k)$ and B_k is a symmetric matrix which is either H_k or an approximation of H_k.

For solving (1), trust region methods usually compute d_k by solving the quadratic sub-problem:

$$\min \ m_k(d) = f_k + g_k^T d + \frac{1}{2} d^T B_k d, \quad \|d\| \le \Delta_k. \tag{2}$$

$\Delta_k > 0$ is a trust region radius. The initial and the updating rule of Δ_k are crucial for the performance of the traditional trust region methods [1-3]. Furthermore, it is obvious that the radius Δ_k in (2) is independent from any information about g_k and B_k. These facts may increase the number of sub-problems that need solving and decrease the efficiency of trust region methods. In order to reduce the number of solving sub-problems, Zhang et al. proposed a strategy to determine the trust region radius [1]. Specifically, they solved the sub-problem (2) with

$$\Delta_k = c^p \|g_k\| \|\hat{B}_k^{-1}\|,$$

where $c \in (0,1)$, p is a nonnegative integer and $\hat{B}_k = B_k + iE$ is a positive definite matrix for some i. Their method utilizes the information of g_k and B_k, however, it needs to estimate $\|\hat{B}_k^{-1}\|$ at each iteration which leads us to some additional computations. Inspired by Zhang's method, Shi et al. [4] proposed a simple adaptive trust region method, in which the Δ_k was computed by the following formula:

$$\Delta_k = c^p \| g_k \|^3 / g_k^T \hat{B}_k g_k , \tag{3}$$

where $c \in (0,1)$, $\hat{B}_k = B_k + iE$ is a positive definite matrix and p is a nonnegative integer.

Besides, Mo et al. [15] proposed a non-monotone trust region method with fixed step-size. In their algorithm, the step-size is computed by a fixed formula if the trial step is rejected. Thus, it can reduce the number of solving sub-problems efficiently. The fixed step-size formula was defined by the following equation:

$$\alpha_k = - \frac{\delta g_k^T d_k}{d_k^T B_k d_k}. \tag{4}$$

2. Non-monotone technique and our strategy

Recently, non-monotone techniques are widely used in the line search and trust region methods. In 1982, the first non-monotone technique that is the so-called watchdog technique was proposed by Chamberlain et al. [5] for constrained optimization to overcome the Maratos effect. Motivated by this idea, Grippo et al. first introduced a non-monotone line search technique for Newton's method in [6]. In 1993, Deng et al. [7] proposed a non-monotone trust region algorithm in which they combined non-monotone term and trust region method for the first time. Due to the high efficiency of non-monotone techniques, many authors are interested in working on the non-monotone techniques for solving optimization problems [8-11]. Especially, nowadays some researchers are focused on utilizing non-monotone techniques in adaptive trust region method and good numerical results have been achieved [12-14].

The general non-monotone form is as follows:

$$f_{l(k)} = f(x_{l(k)}) = \max_{0 \le j \le m(k)} \{ f_{k-j} \}, \quad k = 0, 1, 2, \ldots \tag{5}$$

where $m(0) = 0,\ 0 \le m(k) \le \min\{M, m(k-1)+1\}$ and $M \ge 0$ is an integer constant. Actually, the most common non-monotone ratio is defined as follows:

$$r_k = \frac{f_{l(k)} - f(x_k + d_k)}{m_k(0) - m_k(d_k)}.$$

Some researchers showed that utilizing non-monotone techniques may improve both the possibility of finding the global optimum and the rate of convergence [6, 16]. However, although the non-monotone technique has many advantages, Zhang et al. [16] found that it still has some drawbacks and they proposed a new non-monotone form C_k. Recently, Gu et al. [10] introduced another non-monotone form in 2008 and the new form was computed easier than C_k. They define

$$D_k = \begin{cases} f(x_k) & k = 1; \\ \eta_k D_{k-1} + (1 - \eta_k) f(x_k) & k \ge 2 \end{cases} \tag{6}$$

for some fixed $\eta \in (0,1)$, or a variable η_k. At the same time, they have the new non-monotone ratio:

$$r_k = \frac{D_k - f(x_k + d_k)}{m_k(0) - m_k(d_k)}. \tag{7}$$

Inspired by [4, 10, 15], we use (3), (4) and (6) to present a new non-monotone self-adaptive trust region method with fixed step-size. To be specific, the algorithm first solve sub-problem (2) to compute the trial step d_k, if the trial step is accepted, set $x_{k+1} = x_k + d_k$. Otherwise, the algorithm generates an iterative point whose step length is defined by (4) instead of resolving the sub-problem, i.e. $x_{k+1} = x_k + \alpha_k d_k$. What's more, our algorithm can automatically adjust Δ_k of related sub-problems in the each iteration.

The rest of this paper is organized as follows. In Section 3, we introduce the algorithm of non-monotone self-adaptive trust region method with fixed step-size. In Section 4, we analyze the new method and prove the global convergence. Some conclusions are given in Section 5.

3. New algorithm

In this paper, we consider the following assumptions that will be used to analyze the convergence properties of the below new algorithm (similar to [15]):

(H1) The level set $L_1 = \{ x \in R^n \mid f(x) \le f(x_1) \} \subset \Omega$, where $\Omega \subset R^n$ is a closed, bounded set.

(H2) There exists a constant $v > 0$ such that $d^T B_k d \ge v\|d\|^2$ for all $d \in R^n$.

(H3) $\nabla f(x)$ is a Lipschitz continuous function, i.e. there exists a constant $L > 0$ such that $\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|, \ \forall x, y \in R^n$.

(H4) The constant δ in the following algorithm should satisfy $\delta \in (0, \min\{1, v/L\})$.

The new algorithm can be described as follows:

Algorithm 0

Step 1 An initial point $x_0 \in R^n$ and a symmetric matrix $B_0 \in R^{n\times n}$ are given. The constants $0 < \mu < 1,\ 0 < \delta < 1$, $0 < c < 1,\ 0 < \eta < 1,\ M > 0,\ \Delta_0 > 0,\ \tau > 0$ and $\varepsilon > 0$ are also given. Compute $f(x_0)$ and set $k = 0$.

Step 2 Compute g_k. If $\|g_k\| \le \varepsilon$ then stop, else go to Step 3.

Step 3 Similar to [17], solve (2) inaccurately to determine d_k, satisfying

$$m_k(0) - m_k(d_k) \ge \tau\|g_k\|\min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}, \tag{8}$$

$$g_k^T d_k \le -\tau\|g_k\|\min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}. \tag{9}$$

Step 4 Compute D_k and ρ_k. If $\rho_k \ge \mu$, set $x_{k+1}=x_k + d_k$. Otherwise, compute the step length α_k according to (4), then set $x_{k+1}=x_k + \alpha_k d_k$.

Step 5 Update Δ_{k+1} on the basis of (3), go to step 6.

Step 6 Update the symmetric matrix B_k by a quasi-Newton Formula (such as DFP and BFGS formula), set $k = k+1$, go to step 2.

4. Convergence analysis

For the convenience of expression, we Let $I = \{k \mid \rho_k \ge \mu\}$ and $J = \{k \mid \rho_k < \mu\}$. We need the following lemmas in order to prove the convergence of the new algorithm.

Lemma 1(See Lemma 3.1 in [15]) Suppose that (H2), (H3) and (H4) hold, and Algorithm 0 generates an infinite sequence $\{x_k\}$. Then for all $k \in J$, we have

$$f_{k+1} - f_k \le \frac{\delta}{2}\left(1 - \frac{L\delta}{v}\right) g_k^T d_k \le 0. \tag{10}$$

Lemma 2 Assume that Algorithm 0 generates an infinite sequence $\{x_k\}$. Then we have

$f_{k+1} \le D_{k+1} \le D_k, \quad \forall k \in \infty.$

Proof. From the definition of D_k, we have $D_{k+1} - f_{k+1} = \eta(D_k - f_{k+1})$ and

$$D_{k+1} - D_k = (1-\eta)(f_{k+1} - D_k). \tag{11}$$

We consider two cases:

Case1. $k \in I$. From (7) and (8), we have

$$D_k - f_{k+1} \ge \mu[m_k(0) - m_k(d_k)] \ge \mu\tau\|g_k\|\min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\} \ge 0. \tag{12}$$

Therefore,

$$D_{k+1} - f_{k+1} = \eta(D_k - f_{k+1}) \ge 0, \tag{13}$$

and $D_{k+1} - D_k = (1-\eta)(f_{k+1} - D_k) \le 0$.

Case2. $k \in J$.

If $k-1\in I$, then from (10) and (13), we have $f_{k+1}\le f_k \le D_k$.
If $k-1\in J$, let $M=\{i|1<i\le k, k-i\in I\}$. If $M=\emptyset$, then from (6) and Lemma 1, we have $f_{k+1}\le f_k\le \cdots \le f_1 = D_1$. Now we will use mathematical induction to prove $D_{k+1}\le D_k$.
For $k=1$, $D_2=\eta D_1+(1-\eta)f_2\le \eta f_1+(1-\eta)f_1=f_1=D_1$. For $k=n$, we suppose that we have $D_{n+1}\le D_n$. Then for $k=n+1$, $D_{n+2}=\eta D_{n+1}+(1-\eta)f_{n+2}\le \eta D_n+(1-\eta)f_{n+1}=D_{n+1}$. So we get $D_{k+1}\le D_k$. From (11) and $0<\eta<1$, we know $f_{k+1}\le D_k$. Thus,

$$D_{k+1}=\eta D_k+(1-\eta)f_{k+1}\ge \eta f_{k+1}+(1-\eta)f_{k+1}=f_{k+1}. \tag{14}$$

On the other hand, if $M\ne\emptyset$, let $m=\min\{i\,|\,i\in M\}$. Then from Lemma 1, we have $f_{k+1}\le f_k\le\cdots\le f_{k-m+1}$. Obviously, $k-m\in I$, then we can get $f_{k-m+1}\le D_{k-m+1}\le D_{k-m}$ from Case 1. Thus,
$D_{k-m+2}=\eta D_{k-m+1}+(1-\eta)f_{k-m+2}\le \eta D_{k-m}+(1-\eta)f_{k-m+1}=D_{k-m+1}$. By the induction principle, we have $D_{k+1}\le D_k$. Then we can get (14) again.
Both Case 1 and Case 2 imply that $f_{k+1}\le D_{k+1}\le D_k$. So the proof is finished.

Lemma 3 Suppose that (H1) holds and the sequence $\{x_k\}$ is generated by Algorithm 0. Then, the sequence $\{D_k\}$ is convergent.
Proof. Lemma 2 together with (H1) imply that

$$\exists\lambda \quad s.t. \; \forall n\in\mathbb{N}: \lambda \le f_{k+1}\le D_{k+1}\le D_k\le\cdots\le D_1\le f_1.$$

This shows that the sequence $\{D_k\}$ is convergent.

Lemma 4 Suppose that (H2)-(H4) hold and the Algorithm 0 generates an infinite sequence $\{x_k\}$. Then for all $k\in\mathbb{N}$, there exists a constant $\varphi>0$ such that

$$D_{k+1}\le D_k-(1-\eta)\varphi\,\|g_k\|\min\left\{\Delta_k,\frac{\|g_k\|}{\|B_k\|}\right\},$$

where $\varphi=\min\left\{\mu\tau,\frac{\delta\tau}{2}\left(1-\frac{L\delta}{v}\right)\right\}$.
Proof. We still consider two cases:
Case1. $k\in I$. From (12), we can obtain that

$$f_{k+1}\le D_k-\mu\tau\,\|g_k\|\min\left\{\Delta_k,\frac{\|g_k\|}{\|B_k\|}\right\}.$$

Case2. $k\in J$. From Lemma1, Lemma 2 and (9), we have

$$\begin{aligned} f_{k+1}&\le f_k+\frac{\delta}{2}\left(1-\frac{L\delta}{v}\right)g_k^T d_k\\ &\le D_k-\frac{\delta\tau}{2}\left(1-\frac{L\delta}{v}\right)\|g_k\|\min\left\{\Delta_k,\frac{\|g_k\|}{\|B_k\|}\right\}.\end{aligned}$$

Let $\varphi=\min\left\{\mu\tau,\frac{\delta\tau}{2}\left(1-\frac{L\delta}{v}\right)\right\}$, we can conclude

$$f_{k+1}\le D_k-\varphi\,\|g_k\|\min\left\{\Delta_k,\frac{\|g_k\|}{\|B_k\|}\right\}. \tag{15}$$

Considering (6) and (15), we obtain for all k

$$D_{k+1} = \eta D_k + (1-\eta) f_{k+1}$$
$$\le \eta D_k + (1-\eta)\left(D_k - \varphi \, \| g_k \| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}\right)$$
$$= D_k - (1-\eta)\varphi \, \| g_k \| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}.$$

Lemma 5 Suppose that (H1)-(H4) hold, if there exists a constant $\varepsilon > 0$ such that $\|g_k\| \ge \varepsilon$, then for all $k \in \infty$, we have

$$\lim_{k \to \infty} \min\{\Delta_k, \varepsilon / M_k\} = 0, \tag{16}$$

where $M_k = 1 + \max_{1 \le i \le k} \| B_k \|$.

Proof. From Lemma 4 and the definition of M_k, we have

$$D_{k+1} - D_k \le -(1-\eta)\varphi \varepsilon \min\{\Delta_k, \varepsilon / M_k\}. \tag{17}$$

Using the above inequality and Lemma 3, we have (16) holds immediately.

Lemma 6 (See Lemma 3.7 in [15]) Suppose that (H1)-(H4) hold and $\|g_k\| \ge \varepsilon$ is satisfied for all $k \in \infty$, then for all sufficiently large $k \in J$, we have

$$\| d_k \| \ge \min\{1, \tau \varepsilon (1-\mu)\} / M_k .$$

Lemma 7 Suppose that (H1)-(H4) hold and $\|g_k\| \ge \varepsilon$ is satisfied for all $k \in \infty$, then for all sufficiently large k, there exists a constant $c_1 \in (0,1)$ such that

$$\Delta_k \ge c_1 \min\{1, \tau \varepsilon (1-\mu)\} / M_k .$$

Proof. The proof is similar to Lemma 3.8 in [15], we omit it for convenience.

Theorem 8 Suppose that (H1)-(H4) hold and $\{B_k\}$ satisfies

$$\sum_{k=0}^{+\infty} \frac{1}{M_k} = +\infty . \tag{18}$$

Then sequence $\{x_k\}$ generated by Algorithm 0 satisfies

$$\lim_{k \to \infty} \inf \| g_k \| = 0.$$

Proof. Assume that (18) does not hold, then for all $k \in \infty$, there exists a constant $\varepsilon > 0$ such that $\|g_k\| \ge \varepsilon$. From Lemma 7, we have

$$\min\{\Delta_k, \varepsilon / M_k\} \ge \gamma / M_k , \tag{19}$$

where $\gamma = \min\{c_1, c_1 \tau \varepsilon (1-\mu), \varepsilon\} = \min\{c_1, c_1 \tau \varepsilon (1-\mu)\}$.

From (17) and (19), we have

$$\sum_{k=1}^{\infty} (D_k - D_{k+1}) \ge \sum_{k=1}^{\infty} (1-\eta)\varphi \varepsilon \min\{\Delta_k, \varepsilon / M_k\} \ge \sum_{k=1}^{\infty} (1-\eta)\varphi \varepsilon \gamma / M_k .$$

Using the above inequality and Lemma 3, we have

$\sum_{k=0}^{+\infty} \frac{1}{M_k} \le \frac{1}{(1-\eta)\varphi \varepsilon \gamma} \sum_{k=1}^{\infty} (D_k - D_{k+1}) < \infty$. This contradicts (18). The proof is completed.

5. Conclusions

In this paper, we introduce the algorithm of new non-monotone self-adaptive trust region method with fixed step-size for unconstrained optimization problems based on (3), (4) and (6). When compared with (5), it is obviously that we fully employ the current objective function value f_k. Besides, with the help of adaptive trust region radius (3) and fixed step-size (4), our algorithm can reduce the number of ineffective iterations so that we can decrease the amount of solving sub-problems. We analyzed and proved the global convergence theory under some mild conditions.

Acknowledgments

This work is supported by the national natural science foundation of China (61473111) and the Natural Science Foundation of Hebei Province (Grant No. A2014201003, A2014201100).

References

1. Zhang, X, Zhang J, Liao L; An adaptive trust region method and its convergence. Science in China Series A: Mathematics,2002; 45(5): 620-631.
2. Zhang JL, Zhang XS; A nonmonotone adaptive trust region method and its convergence. Computers & Mathematics with Applications,2003; 45(10): 1469-1477.
3. Ataee Tarzanagh D, Peyghami MR, Mesgarani H; A new nonmonotone trust region method for unconstrained optimization equipped by an efficient adaptive radius. Optimization Methods and Software, 2014; 29(4): 819-836.
4. Shi ZJ, Wang HQ; A new self-adaptive trust region method for unconstrained optimization, Technical Report, College of Operations Research and Management, Qufu Normal University, 2004.
5. Chamberlain RM, Powell MJD; The watchdog technique for forcing convergence in algorithm for constrained optimization, Mathematical Programming Study, 1982; 1-17.
6. Grippo L, Lampariello F, Lucidi S; A nonmonotone line search technique for Newton's method, Society for Industrial and Applied Mathematics, 1986; 23(4): 707-716.
7. Deng NY, Xiao Y, Zhou FJ; Nonmonotone trust region algorithm, Journal of Optimization Theory and Application,1993; 76(2): 259-285.
8. Toint PL; Non-monotone trust-region algorithm for nonlinear optimization subject to convex constraints, Mathmatical Programming, 1997; 77(3): 69-94.
9. Sun WY; Nonmonotone trust region method for solving optimization problems, Applied Mathematics and Computation, 2004; 156(1): 159-174.
10. Gu NZ, Mo JT; Incorporating nonmonotone strategies into the trust region for unconstrained optimization, Computers and Mathematics with Applications, 2008; 55(9): 2158–2172.
11. Ahookhosh M, Amini K, Bahrami S; A class of nonmonotone Armijo-type line search method for unconstrained optimization, Optimization, 2012; 61(4): 387-404.
12. Yang Y, Sun WY; A new nonmonotone self-adaptive trust region algorithm with line search, Chinese Journal of Engineering Mathematics, 2007; 24(5): 788-794.
13. Fu JH, Sun WY: Nonmonotone adaptive trust-region method for unconstrained optimization problems, Applied Mathematics and Computation, 2005; 163(1): 489-504.
14. Shi ZJ, Guo JH; Nonmonotone adaptive trust region method, European Journal of Operational Research, 2011; 208 (1): 28-36.
15. J.T. Mo, Liu C, Yan SC; A nonmonotone trust region method with fixed stepsize, Journal of Qufu Normal University, 2006; 32(3): 30-34.
16. Zhang HC, Hager WW; A nonmonotone line search technique and its application to unconstrained optimization, SIAM J. Optim, 2004; 14 (4): 1043-1056.
17. Nocedal J, Yuan YX; Combining trust region and line search techniques, in: Y. Yuan (Ed.), Advanced in nonlinear programming, Kluwer Academic Publishers, Dordrecht, 1998; 153-175.