CONVERGENCE ANALYSIS OF ADAPTIVE TRUST REGION METHODS

ZHENG-JUN SHI\(^1\), XIANG-SUN ZHANG\(^2\) AND JIE SHEN\(^3\)

Abstract. In this paper, we propose a new class of adaptive trust region methods for unconstrained optimization problems and develop some convergence properties. In the new algorithms, we use the current iterative information to define a suitable initial trust region radius at each iteration. The initial trust region radius is more reasonable in the sense that the trust region model and the objective function are more consistent at the current iterate. The global convergence, super-linear and quadratic convergence rate are analyzed under some mild conditions. Numerical results show that some special adaptive trust region methods are available and efficient in practical computation.

Keywords. Adaptive trust region method, unconstrained optimization, global convergence, super-linear convergence.

Mathematics Subject Classification. 90C30, 49M37, 65K05.

1. INTRODUCTION

Trust region method is an important technique for solving optimization problems and has wide applications in many fields, such as science, engineering, economy and operations research, etc., due to its strong global convergence and robustness [2–4,16,21,24,29,30].

\(^1\) College of Operations Research and Management, Qufu Normal University, Rizhao, Shandong 276826, P.R. China, and Department of Computer & Information Science, University of Michigan-Dearborn, Michigan MI48128, USA; zjshi@umd.umich.edu or zjshi@qrnu.edu.cn

\(^2\) Institute of Applied Mathematics, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, P.O. Box 2734, Beijing 100080, P.R. China; zxs@amt.ac.cn

\(^3\) Department of Computer & Information Science, University of Michigan-Dearborn, Michigan MI48128, USA; shen@umich.edu

\(\copyright\) EDP Sciences, ROADEF, SMAI 2007

Received March 3, 2005. Accepted October 16, 2006.

* The work was supported in part by NSF DMI-0514900, USA.
Consider an unconstrained optimization problem

$$\min f(x), x \in \mathbb{R}^n,$$

where \( \mathbb{R}^n \) is an n-dimensional Euclidean space and \( f : \mathbb{R}^n \to \mathbb{R}^1 \) is a twice continuously differentiable function.

Throughout this paper we denote

- \( x_k \) as the current iterate (\( k = 0, 1, 2, \ldots \));
- \( f_k = f(x_k), g_k = g(x_k) = \nabla f(x_k), H_k = H(x_k) = \nabla^2 f(x_k) \);
- \( \| \cdot \| \) as the Euclidean norm;
- \( B_k \) as a symmetric matrix and an approximation to \( H_k \) in some sense at the point \( x_k \);
- \( \hat{B}_k = B_k + iI \) (\( I \in \mathbb{R}^{n \times n} \) denotes the unit matrix), \( i \) is the smallest nonnegative integer such that \( \hat{B}_k \) becomes a positive definite matrix.

The existing methods for solving (1) can be divided into two classes: one is line search method and the other is trust region method. Line search method needs to carry out a line search procedure at each iteration and trust region method needs to solve a trust region subproblem at each step. As we have known, trust region methods are based on the following approach. At the iterate \( x_k \) (suppose that it is not a stationary point), a trial step is usually obtained by solving the following subproblem

$$\min_{d \in \mathbb{R}^n} m_k(d) = g_k^T d + \frac{1}{2} d^T B_k d, \quad s.t. ~ \|d\| \leq \Delta_k,$$

where \( \Delta_k \) is a trust region radius. A merit function is normally used to test whether the trial step is accepted or the trust region radius needs to be adjusted. In comparison with quasi-Newton methods, trust region methods can converge to such a point that is not only a stationary point, but also satisfies second-order necessary conditions. Because of its strong convergence and robustness, trust region methods have been studied by many authors \([1, 5, 9–12, 17, 19, 22, 25]\) and some convergence properties are given in the literature \([6–8, 14, 15, 18, 20, 26]\).

It is well known that the trust region radius \( \Delta_k \) in the above mentioned subproblem is independent of \( g_k \) and \( B_k \). As a result, we do not know whether the quasi-Newton step, \(- B_k^{-1} g_k\), is feasible at the \( k \)-th step, even when the test condition of the merit function is satisfied. This situation would decrease the efficiency of these methods. Furthermore, the choice of \( \Delta_k \) also affects the efficiency of these methods.

Sartenaer \([19]\) presented a strategy for determining automatically an initial trust region radius. The basic idea is to determine a maximal initial radius through many repeated trials in the direction, \(- g_k\), in order to guarantee a sufficient agreement between the model and the objective function. Zhang et al. \([27]\) presented another strategy of determining the trust region radius. Their basic idea originated
from the following subproblem in an artificial neural network research [25, 26],

\[
min \quad m_k(d) = g_k^T d + \frac{1}{2} d^T B_k d \\
s.t. \quad -\alpha(x_k) \leq d_i \leq \alpha(x_k), \quad i = 1, 2, ..., n,
\]

where \( \alpha_k = c p \frac{\|g_k\|}{m_k} \), \( 0 < c < 1 \), \( m_k = \min(\|B_k\|, 1) \), and \( p \) is a nonnegative integer. In their algorithm, instead of adjusting \( \Delta_k \), they adjust \( p \) at each iterate. Motivated by this technique, they solved the trust region subproblem (2) with

\[
\Delta_k = c^p \|g_k\| \cdot \|\hat{B}_k^{-1}\| \tag{3}
\]

and gave a global convergent adaptive trust region method [28], where \( c \in (0, 1) \) and \( p \) is a nonnegative integer.

However, their method needs to estimate \( \|B_k\| \) or \( \|\hat{B}_k^{-1}\| \) at each iteration, which leads to some additional cost of computation. As a result, a simple adaptive trust region method was proposed [23], which used the following trust region radius

\[
\Delta_k = c^p \|g_k\|^3 / g_k^T \hat{B}_k g_k \tag{4}
\]

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

In this paper, we present a new class of adaptive trust region methods for unconstrained optimization problems and develop several convergence properties. At each iteration, the new methods generate a suitable initial trust region radius automatically based on the current iterative information. The new trust region model is more consistent with the objective function at the current iterate. The global convergence and super-linear and quadratic convergence rate of these new methods are proved under some mild conditions. Numerical results show that some special adaptive trust region methods are available and efficient in practical computation.

The rest of this paper is organized as follows. In the next section, we introduce the new class of adaptive trust region methods and give some simple properties. In Sections 3 and 4, the global convergence and super-linear and quadratic convergence rate are investigated. Numerical results are given in Section 5. Conclusions and future research are summarized in Section 6.

2. NEW ALGORITHMS AND SOME PROPERTIES

The new trust region methods for (1) at the current iterate \( x_k \) need to solve the subproblem

\[
\min_{d \in \mathbb{R}^n} m_k(d) = g_k^T d + \frac{1}{2} d^T B_k d \quad s.t. \quad \|d\| \leq \alpha_k \tag{5}
\]

where

\[
\alpha_k = -c^p \frac{g_k^T q_k}{q_k^T \hat{B}_k q_k} \|q_k\|, \tag{6}
\]
(0 < c < 1), p is a nonnegative integer and \( q_k \) satisfies
\[
- \frac{q_k^T q_k}{\|g_k\| \cdot \|q_k\|} \geq \tau
\]  \( (7) \)
with \( \tau \in (0, 1] \).

We can determine \( \hat{B}_k \) by carrying out the procedure: if \( q_k^T B_k q_k > 0 \) then \( \hat{B}_k = B_k \) else test \( \hat{B}_k = B_k + iI \) for \( i = 1, 2, 3, \cdots \) until \( q_k^T \hat{B}_k q_k = q_k^T B_k q_k + i\|q_k\|^2 > 0 \).

Define the following parameter
\[
Pred_k = m_k(0) - m_k(d_k) = -m_k(d_k) = - \left( g_k^T d_k + \frac{1}{2} d_k^T B_k d_k \right),
\]
\[
Ared_k = f_k - f(x_k + d_k),
\]
\[
r_k = \frac{Ared_k}{Pred_k}
\]
We assume that
\( (H1) \). The objective function \( f(x) \) is twice continuously differentiable on \( R^n \) and the level set \( L(x_0) = \{ x \in R^n | f(x) \leq f(x_0) \} \) is bounded for a given \( x_0 \).

\( (H2) \). \( \{B_k\} \) is uniformly bounded, i.e. there exists an \( M \) such that \( \|B_k\| \leq M \) for all \( k \).

**Remark 2.1.** Since \( f(x) \) is a twice continuously differentiable function, \( (H1) \) implies that \( \{\nabla^2 f(x)\} \) is uniformly continuous and bounded on a bounded open convex set \( \Omega \) that contains \( L(x_0) \). Hence, there exists \( L \) such that \( \|\nabla^2 f(x)\| \leq L \) and
\[
\|g(x) - g(y)\| \leq L\|x - y\|, \ \forall x, y \in \Omega.
\]

**Remark 2.2.** By the generating procedure of \( \hat{B}_k \) and \( (H2) \), we obtain that \( \{\hat{B}_k\} \) is also uniformly bounded. In fact, assume that \( \|B_k\| \leq M_0 \) for all \( k \) with \( M_0 \) being a positive constant, then \( \|\hat{B}_k\| = \|B_k + iI\| \leq 2M_0 \) whenever \( q_k^T \hat{B}_k q_k = q_k^T B_k q_k + i\|q_k\|^2 > 0 \).

**Algorithm (A).**

**Step 0.** Set \( 0 < c < 1, \ \epsilon \geq 0, \ 0 < \eta < 1, x_0 \in R^n \) and \( p := 0, k := 0 \).

**Step 1.** If \( \|q_k\| \leq \epsilon \) then stop else solve (5) to obtain \( d_k \). Set \( \hat{x}_{k+1} = x_k + d_k \) and go to Step 2.

**Step 2.** If \( r_k < \eta \) then set \( p := p + 1 \) and go to Step 1.

**Step 3.** Set \( x_{k+1} = \hat{x}_{k+1} \), modify \( B_k \) as \( B_{k+1} \), set \( p := 0, k := k + 1 \) and go to Step 1.

**Lemma 2.3.** For \( k \geq 1 \),
\[
Pred_k \geq \frac{c^p (q_k^T q_k)^2}{2q_k^T B_k q_k}, \ \ \forall p = 0, 1, 2, \cdots
\]
Proof. Take $d = -\frac{c^p g_k^T q_k}{q_k^T B_k q_k} q_k$, and since

$$
||d|| = -\frac{c^p g_k^T q_k}{q_k^T B_k q_k} ||q_k||
= \alpha_k,
$$

it follows that $d$ is a feasible solution to (5). Noting that $d_k$ is an optimal solution to (5), we have $m_k(d_k) \leq m_k(d)$, and thus,

$$
Pred_k \geq \text{Pred}
= -(g_k^T d + \frac{1}{2} d^T B_k d)
= - \left( \frac{c^p(g_k^T q_k)^2}{q_k^T B_k q_k} + \frac{c^{2p}(g_k^T q_k)^2 q_k^T B_k q_k}{2(q_k^T B_k q_k)^2} \right)
\geq - \left( \frac{c^p(g_k^T q_k)^2}{q_k^T B_k q_k} + \frac{c^{2p}(g_k^T q_k)^2 q_k^T B_k q_k}{2(q_k^T B_k q_k)^2} \right)
= \frac{c^p(g_k^T q_k)^2}{2q_k^T B_k q_k}.
$$

Lemma 2.4. For $k \geq 1$, it holds that

$$
Pred_k \geq \frac{1}{2} \left( -\frac{g_k^T q_k}{||q_k||} \right) \min \left\{ \alpha_k, -\frac{g_k^T q_k}{q_k^T B_k q_k} ||q_k|| \right\}.
$$

Proof. In the case of $\alpha_k < -\frac{g_k^T q_k}{q_k^T B_k q_k} ||q_k||$, since $d = -\frac{c^p g_k^T q_k}{q_k^T B_k q_k} q_k$ is a feasible solution to (5) and $d_k$ is an optimal solution to (5), we have $m_k(d_k) \leq m_k(d)$, and thus,

$$
Pred_k \geq \text{Pred}
= -(g_k^T d + \frac{1}{2} d^T B_k d)
= - \left( \frac{c^p(g_k^T q_k)^2}{g_k^T B_k q_k} + \frac{c^{2p}(g_k^T q_k)^2 q_k^T B_k g_k}{2(g_k^T B_k g_k)^2} \right)
\geq - \left( \frac{c^p(g_k^T q_k)^2}{g_k^T B_k q_k} + \frac{c^{2p}(g_k^T q_k)^2 q_k^T B_k g_k}{2(g_k^T B_k g_k)^2} \right)
= \frac{c^p(g_k^T q_k)^2}{2g_k^T B_k g_k}
= -\frac{1}{2} \frac{g_k^T q_k}{||q_k||} \alpha_k.
$$
In the case of $\alpha_k \geq -\frac{g_k^T q_k}{q_k^T \hat{B}_k q_k} \|q_k\|$, since $d = -\frac{g_k^T q_k}{q_k^T \hat{B}_k q_k} q_k$ is a feasible solution to (5) and $d_k$ is an optimal solution to (5), we have $m_k(d_k) \leq m_k(d)$, and thus,

$$\text{Pred}_k \geq \frac{g_k^T q_k}{q_k^T \hat{B}_k q_k} \min \left\{ \alpha_k, \frac{g_k^T q_k}{g_k^T \hat{B}_k q_k} \right\}. \quad \square$$

Lemma 2.5. If (H1) and (H2) hold, then Algorithm (A) is well-defined, i.e. Algorithm (A) does not circle at Step 1–Step 2 infinitely.

Proof. Suppose that Algorithm (A) circles at Step 1–Step 2 infinitely and denotes $\text{Pred}_k \geq \text{Pred}_{k(i)} \geq \text{Pred}_{k(i)}^{\text{Pred}_k} \geq \text{Pred}_{k(i)}^{\text{Pred}_k}$, where $x_{k(i)} = x_k, p_k(i) = i$, and $r_{k(i)} < \eta, \ i = 0, 1, 2, \ldots$. Lemma 2.3 shows that

$$\text{Pred}_{k(i)} \geq \frac{\epsilon^{p_k(i)}(g_k^T q_k)^2}{2q_k^T \hat{B}_k q_k} \forall i = 0, 1, 2, \ldots \quad (8)$$

By (H1) and the definition of $\text{Pred}_k$ and $A_{\text{red}}$, we have

$$A_{\text{red}} - \text{Pred}_k = O(||d_k||^2). \quad (9)$$

By (8) and (9) we obtain

$$|r_{k(i)} - 1| = \frac{|A_{\text{red}}(i) - \text{Pred}_k(i)|}{|\text{Pred}_k(i)|} \leq \frac{O(||d_k(i)||^2)}{\epsilon^{p_k(i)}(g_k^T q_k)^2} \frac{2q_k^T \hat{B}_k q_k}{2q_k^T \hat{B}_k q_k}$$

$$\leq \frac{O(||\alpha_k(i)||^2)}{\epsilon^{p_k(i)}(g_k^T q_k)^2} \frac{\epsilon^{p_k(i)}2||q_k||^2}{\epsilon^{p_k(i)}2||q_k||^2}$$

$$\rightarrow 0 \ (i \rightarrow +\infty).$$
and thus, for sufficiently large $i$, we have $r_k(i) \geq \eta$. This contradicts $r_k(i) < \eta$ and the proof is completed. \hfill \Box

3. Global convergence

**Theorem 3.1.** If (H1) and (H2) hold, $\epsilon = 0$ and $g_k^T q_k < 0$, then Algorithm (A) either stops at a stationary point of (1) in finite step or generates an infinite sequence $\{x_k\}$ such that

$$\lim_{k \to \infty} - (g_k^T q_k / \|q_k\|) = 0. \quad (10)$$

**Proof.** Suppose that Algorithm (A) generates an infinite sequence $\{x_k\}$ and $\lim_{k \to \infty} - (g_k^T q_k / \|q_k\|) \neq 0$. Then there exist $\epsilon_0 > 0$ and an infinite subset $K \subseteq \mathbb{N} = \{0, 1, 2, \ldots\}$ such that

$$- \frac{g_k^T q_k}{\|q_k\|} \geq \epsilon_0, \quad \forall k \in K. \quad (11)$$

(H2) implies that there exists an $M_0 > 0$ such that

$$\|B_k\| \leq M_0, \forall k \in K. \quad (12)$$

By (11), (12), Lemma 2.3 and Algorithm (A), we have

$$\sum_{k \in K} [f_k - f_{k+1}] \geq \sum_{k \in K} \eta \text{Pred}_k \geq \sum_{k \in K} \eta \epsilon_k^2 \|B_k\| / (2\|q_k\|). \quad (13)$$

where $p^k$ is the largest $p$ at the $k$-th iteration of Algorithm (A). Combining (12) and (13), we have

$$\sum_{k \in K} [f_k - f_{k+1}] \geq \sum_{k \in K} \frac{\eta \epsilon_k^2}{2M_0} \epsilon^k.$$ 

Since $\{f(x_k)\}$ decreases monotonically and has a bound from below, we obtain

$$\frac{\eta \epsilon_k^2}{2M_0} \epsilon^k \to 0, (k \in K, \ k \to +\infty).$$

Therefore, $p^k \to +\infty (k \in K, \ k \to +\infty)$ and we can assume that $p^k \geq 1, \forall k \in K$. Algorithm (A) shows that the solution $d_k$ to the subproblem

$$\min_{d \in \mathbb{R}^n} m_k(d) = g_k^T d + \frac{1}{2} d^T B_k d$$

s.t. $\|d\| \leq -c^{p^k-1} \frac{g_k^T q_k}{g_k^T B_k g_k} \|q_k\|$, \quad ($0 < c < 1$),
cannot be accepted by the algorithm. That is to say, if \( \tilde{x}_{k+1} = x_k + \tilde{d}_k \) then we have

\[
\frac{f(x_k) - f(\tilde{x}_{k+1})}{-m_k(\tilde{d}_k)} < \eta. \tag{14}
\]

On the other hand, by (12), Lemma 2.4 and Algorithm (A), we obtain

\[
\sum_{k \in K} [f_k - f_{k+1}] \geq \sum_{k \in K} \eta \text{Pre}_d_k
\]

\[
\geq - \sum_{k \in K} \eta \frac{1}{2} \|q_k\| \min \left\{ \alpha_k, -\frac{\|q_k\| g_k^T q_k}{q_k^T B_k q_k} \right\}
\]

\[
\geq \sum_{k \in K} \eta \frac{c_0}{2} \min \left\{ \alpha_k, -\frac{c_0}{q_k^T B_k q_k} \right\}
\]

\[
\geq \sum_{k \in K} \eta \frac{c_0}{2} \min \left\{ \alpha_k, \frac{c_0}{M_0} \right\}.
\]

By \((H1), (H2)\) and the monotonicity of \(\{f(x_k)\}\), we have

\[
\alpha_k \to 0 (k \in K, k \to +\infty). \tag{15}
\]

Since

\[
f(\tilde{x}_{k+1}) - f(x_k) - m_k(\tilde{d}_k) = O(\|\tilde{d}_k\|^2),
\]

and by Lemma 2.4, (11), (12) and (15), we have

\[
|r_k - 1| = \left| \frac{f(x_k) - f(\tilde{x}_{k+1})}{-m_k(d_k)} - 1 \right|
\]

\[
\leq \frac{O(\|\tilde{d}_k\|^2)}{2} \min \left\{ \alpha_k, \frac{\alpha_k}{\tilde{x}_0} \right\}
\]

\[
\leq \frac{O(\alpha_k^2)}{2c_0 \alpha_k}
\]

\[
\to 0 (k \in K, k \to +\infty),
\]

and thus, for sufficiently large \(k\), we have

\[
\frac{f(x_k) - f(\tilde{x}_{k+1})}{-m_k(\tilde{d}_k)} \geq \eta.
\]

This is a contradiction to (14). Therefore, there is no such an infinite subset \(K\) and \(\epsilon_0 > 0\) such that (11) holds. This shows that (10) holds. \(\square\)

**Corollary 3.2.** If the conditions in Theorem 3.1 hold and \(q_k\) satisfies (7), then

\[
\lim_{k \to \infty} \|g_k\| = 0.
\]
Proof. Since $q_k$ satisfies (7), by Theorem 3.1 we have
\[
\|g_k\|_\tau \leq \frac{-g_k^T q_k}{\|g_k\| \cdot \|q_k\|} \|g_k\| = -\frac{g_k^T q_k}{\|q_k\|} \to 0 (k \to \infty).
\]
This shows the conclusion. \(\square\)

4. Convergence rate

Theorem 4.1. Assume that (H1) and (H2) hold and Algorithm (A) generates an infinite sequence \(\{x_k\}\) such that \(x_k \to x^*(k \to +\infty)\). \(H(x)\) is Lipschitz continuous on a neighborhood \(N(x^*, \epsilon)\) of \(x^*\) and \(H(x^*)\) and \(B_k\) are positive definite matrices such that
\[
\lim_{k \to \infty} \frac{||B_k - H(x^*)||q_k||}{||q_k||} = 0,
\]
where \(q_k = -B_k^{-1} g_k\). Then \(\{x_k\}\) converges to \(x^*\) super-linearly.

Proof. Because \(\dot{B}_k = B_k\) for sufficiently large \(k\), we assert that \(\dot{d}_k = \frac{-g_k^T q_k}{q_k^T B_k q_k} q_k\) is an optimal solution to the subproblem
\[
\min_{d \in \mathbb{R}^n} m_k(d) = \frac{g_k^T d + \frac{1}{2}d^T B_k d}{\|q_k\|}
\]
subject to
\[
\|d\| \leq -\frac{g_k^T q_k}{q_k^T \dot{B}_k q_k} ||q_k||.
\]
In the sequel, we will prove that
\[
\frac{f(x_k) - f(x_k + \dot{d}_k)}{-m_k(\dot{d}_k)} > \eta,
\]
for sufficiently large \(k\). By (16) we have
\[
g_k + H(x^*) \dot{d}_k = o(||\dot{d}_k||),
\]
i.e.
\[
\dot{d}_k = -H(x^*)^{-1} g_k + o(||\dot{d}_k||),
\]
thus
\[
||\dot{d}_k|| \leq ||H(x^*)^{-1}|| \cdot ||g_k|| + o(||\dot{d}_k||),
\]
and consequently
\[
\frac{||g_k||}{||\dot{d}_k||} \geq \frac{1}{||(H(x^*))^{-1}||} + \frac{o(||\dot{d}_k||)}{||\dot{d}_k||}.
\]
By Corollary 3.2 we have $g_k \to 0 \ (k \to +\infty)$, and therefore $\hat{d}_k \to 0 \ (k \to +\infty)$.

By Lemma 2.3 and noting that $-q_k^T q_k = q_k^T B_k q_k$ we have

$$-m_k(\hat{d}_k) \geq \frac{g_k^2 q_k^2}{2q_k^T B_k q_k} \geq \frac{g_k^2 B_k q_k}{2}. \tag{17}$$

By (16) and noting that $q_k = -B_k^{-1} g_k = \hat{d}_k$ we have

$$f(x_k + \hat{d}_k) - f(x_k) - m_k(\hat{d}_k) = o(||\hat{d}_k||^2). \tag{18}$$

By (17), (18) and (12), we have

$$\left| \frac{f(x_k) - f(x_k + \hat{d}_k)}{-m_k(\hat{d}_k)} - 1 \right| \leq \frac{o(||\hat{d}_k||^2)}{q_k^T B_k q_k} \leq \frac{2o(||\hat{d}_k||^2)}{M_0 ||q_k||^2} \leq \frac{2o(||\hat{d}_k||^2)}{M_0 ||d_k||^2} \rightarrow 0 \ (k \to +\infty),$$

which implies that

$$\frac{f(x_k) - f(x_k + \hat{d}_k)}{-m_k(\hat{d}_k)} > \eta,$$

for sufficiently large $k$. Therefore $x_{k+1} = x_k + \hat{d}_k$ for sufficiently large $k$. This shows that Algorithm (A) reduces to a quasi-Newton method for sufficiently large $k$. We can complete the rest proof by citing [2, 5, 15] or other related literature. □

**Theorem 4.2.** Assume that (H1) and (H2) hold and Algorithm (A) generates an infinite sequence $\{x_k\}$ such that $x_k \to x^* \ (k \to +\infty)$. $H(x)$ is Lipschitz continuous on a neighborhood $N(x^*, \epsilon)$ of $x^*$ and $H(x^*)$ and $B_k$ are positive definite matrices with $B_k = H(x_k)$ and $q_k = -B_k^{-1} g_k$. Then $\{x_k\}$ converges to $x^*$ quadratically.

**Proof.** Since $B_k = H(x_k)$ implies that all conditions of Theorem 4.1 hold, we have $\hat{d}_k = q_k \to 0 \ (k \to \infty)$. Therefore, there exists a $k'$ such that

$$x_k + t q_k \in N(x^*, \epsilon), \ k \geq k', \ t \in [0, 1]. \tag{19}$$

Because $H(x)$ is Lipschitz continuous on the neighborhood $N(x^*, \epsilon)$ of $x^*$, there must exist an $L(\epsilon) > 0$ such that

$$\|H(x) - H(y)\| \leq L(\epsilon) \|x - y\|, \ \forall x, y \in N(x^*, \epsilon). \tag{20}$$

In this case, Algorithm (A) reduces to Newton method for sufficiently large $k$. The rest proof can also be seen from [2, 5, 15]. □
Theorem 4.3. Assume that (H1) and (H2) hold and \( B_k = H_k \), \( \alpha_k = -c^p \frac{\|q_k\|^2}{Q_k} \frac{\tilde{q}_k}{Q_k} \), where \( 0 < c < 1 \) and \( p \) is a nonnegative integer. If \( \{x_k\} \) converges to \( x^* \), then \( H(x^*) \) is a semi-positive definite matrix, i.e., \( x^* \) satisfies the second order necessary condition.

Proof. Denote \( \lambda_k^1 \) and \( \lambda^* \) as the smallest eigenvalue of \( H_k \) and \( H(x^*) \) respectively. Let \( z_k \) be a normal eigenvector (\( \|z_k\| = 1 \)) of \( H_k \) corresponding to the eigenvalue \( \lambda_k^1 \) and \( z_k^T g_k \leq 0 \), and then \( H_k z_k = \lambda_k^1 z_k \). Suppose that \( H(x^*) \) is not a positive semi-definite matrix, then \( \lambda^* < 0 \) and thus \( \lambda_k^1 < 0 \) for sufficiently large \( k \).

Because \( \|\alpha_k z_k\| = \alpha_k \), it follows that \( \alpha_k z_k \) is a feasible solution to (5). Therefore,

\[
P_{\text{red}} \geq -\left( \alpha_k g_k^T z_k + \frac{1}{2} \alpha_k^2 z_k^T B_k z_k \right) \geq -\frac{1}{2} \alpha_k^2 \lambda_k^1 \|z_k\|^2 = -\frac{1}{2} \alpha_k^2 \lambda_k^1. \tag{21}
\]

By (21) and Algorithm (A) we have

\[
A_{\text{red}} \geq \eta P_{\text{red}} \geq -\frac{1}{2} \eta \alpha_k^2 \lambda_k^1.
\]

Since \( \{f(x_k)\} \) is a monotone decreasing sequence and has a bound from below, we have

\[
A_{\text{red}} = f_{k+1} - f_k \rightarrow 0 \quad (k \rightarrow +\infty),
\]

and thus \( \alpha_k^2 \lambda_k^1 \rightarrow 0 \quad (k \rightarrow +\infty) \). Noting that \( \lambda_k^1 \rightarrow \lambda^* \quad (k \rightarrow \infty) \), we have

\[
\lim_{k \rightarrow \infty} \alpha_k = 0.
\]

From the definition of Algorithm (A), we can observe that the solution \( \tilde{d}_k \) to

\[
\min_{d \in \mathbb{R}^n} m_k(d) = g_k^T d + \frac{1}{2} d^T B_k d
\]

s.t. \( \|d\| \leq \hat{\alpha}_k \)

with \( \hat{\alpha}_k = -c^{p-1} \frac{\|q_k\|^2}{Q_k} \frac{\tilde{q}_k}{Q_k} \) (\( 0 < c < 1 \)), cannot be accepted by the algorithm, i.e.

if \( \tilde{x}_{k+1} = x_k + \tilde{d}_k \), then

\[
\frac{f(x_k) - f(\tilde{x}_{k+1})}{-m_k(d_k)} < \eta. \tag{22}
\]

On the other hand, noting that \( B_k = H_k \), by Taylor expansion, we have

\[
|f(x_k) - f(\tilde{x}_{k+1}) + m_k(d_k)| = o(||\tilde{d}_k||^2). \tag{23}
\]
By (21) and (23) we obtain

\[ \left| \frac{f(x_k) - f(\tilde{x}_{k+1}) - m_k(\tilde{d}_k) - 1}{-m_k(d_k)} \right| \leq o(\|d_k\|^2) - \frac{1}{2} \tilde{\alpha}_k^2 \lambda_k^1 \]

\[ \leq o(\tilde{\alpha}_k^2) - \frac{1}{2} \tilde{\alpha}_k^2 \lambda_k^1 \]

\[ \to 0(k \to \infty). \]

Since \( \lambda_k^1 \to \lambda^* < 0 \), we get

\[ \left| \frac{f(x_k) - f(\tilde{x}_{k+1}) - m_k(\tilde{d}_k) - 1}{-m_k(d_k)} \right| \to 0(k \to \infty), \]

and thus

\[ \frac{f(x_k) - f(\tilde{x}_{k+1}) - m_k(d_k)}{-m_k(d_k)} \geq \eta \]

for sufficiently large \( k \), which contradicts (22). This shows that \( H(x^*) \) is a positive semi-definite matrix. \( \square \)

5. NUMERICAL RESULTS

In adaptive trust region methods, \( q_k \) satisfying (7) has a wide scope. Of course, \( q_k = -g_k \) is a natural choice, which leads to a recently proposed adaptive trust region method [23]. If we take \( q_k = -\tilde{B}_k^{-1}g_k \), then we can obtain a new adaptive trust region method. In this case, the trust region radius is \( \alpha_k = \epsilon \| \tilde{B}_k^{-1}g_k \| \) at the \( k \)-th step. The adaptive trust region methods with \( q_k = -g_k \) and \( q_k = -\tilde{B}_k^{-1}g_k \) are denoted by TRS and TRN respectively. The original trust region method with \( \bar{\Delta} = 100, \Delta_0 = 50 \) and \( \eta = 0.01 \) is denoted by TRO [15] (p. 68), which is described as follows.

**Algorithm 5.1** (trust region)

Given \( \bar{\Delta} > 0, \Delta_0 \in (0, \bar{\Delta}) \), and \( \eta \in [0, \frac{1}{4}] \):

For \( k = 0, 1, 2, \ldots \):

- Obtain \( d_k \) by (approximately) solving (2);
- Evaluate \( r_k \);
- if \( r_k < \frac{1}{4} \)
  - \( \Delta_{k+1} = \frac{1}{4} \| d_k \| \)
- else
  - if \( r_k > \frac{1}{4} \) and \( \| d_k \| = \Delta_k \)
    - \( \Delta_{k+1} = \min(2\Delta_k, \bar{\Delta}) \)
  - else
    - \( \Delta_{k+1} = \Delta_k \);
Table 1. Iterations, function and gradient evaluations and CPU time.

<table>
<thead>
<tr>
<th>P</th>
<th>n</th>
<th>TRS</th>
<th>TRN</th>
<th>TRI</th>
<th>TRZ</th>
<th>TRO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>26/88/26</td>
<td>18/36/18</td>
<td>28/62/28</td>
<td>26/59/29</td>
<td>36/56/56</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>23/45/23</td>
<td>18/34/18</td>
<td>27/58/27</td>
<td>32/32/32</td>
<td>42/57/52</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>15/46/15</td>
<td>12/23/12</td>
<td>16/52/16</td>
<td>22/33/22</td>
<td>20/78/60</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>15/55/16</td>
<td>12/18/12</td>
<td>17/68/17</td>
<td>34/64/34</td>
<td>28/75/68</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>26/48/26</td>
<td>21/26/21</td>
<td>26/68/26</td>
<td>37/87/37</td>
<td>32/78/72</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>18/49/18</td>
<td>15/15/15</td>
<td>23/63/23</td>
<td>35/38/35</td>
<td>28/85/78</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>43/46/43</td>
<td>36/36/36</td>
<td>52/65/52</td>
<td>61/72/61</td>
<td>48/67/67</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>48/62/48</td>
<td>35/35/35</td>
<td>53/58/53</td>
<td>64/68/64</td>
<td>83/75/72</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>46/56/46</td>
<td>37/37/37</td>
<td>63/68/63</td>
<td>66/74/66</td>
<td>85/85/81</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>46/62/46</td>
<td>38/38/38</td>
<td>53/57/53</td>
<td>59/78/59</td>
<td>72/83/78</td>
</tr>
<tr>
<td>12</td>
<td>3</td>
<td>46/68/46</td>
<td>37/37/37</td>
<td>65/75/65</td>
<td>69/72/69</td>
<td>83/88/83</td>
</tr>
<tr>
<td>13</td>
<td>4</td>
<td>56/58/56</td>
<td>45/45/45</td>
<td>68/74/68</td>
<td>81/89/81</td>
<td>91/95/91</td>
</tr>
<tr>
<td>14</td>
<td>4</td>
<td>38/45/38</td>
<td>29/29/29</td>
<td>43/48/43</td>
<td>68/78/68</td>
<td>86/92/86</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>76/76/76</td>
<td>59/59/59</td>
<td>85/88/85</td>
<td>97/99/97</td>
<td>94/98/94</td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>29/36/29</td>
<td>18/18/18</td>
<td>43/63/43</td>
<td>78/84/78</td>
<td>93/98/93</td>
</tr>
<tr>
<td>17</td>
<td>5</td>
<td>93/98/93</td>
<td>79/79/79</td>
<td>98/98/98</td>
<td>85/96/85</td>
<td>92/123/93</td>
</tr>
<tr>
<td>18</td>
<td>6</td>
<td>38/46/38</td>
<td>26/26/26</td>
<td>38/64/38</td>
<td>54/59/54</td>
<td>63/82/78</td>
</tr>
</tbody>
</table>

CPU – 118 s 59 s 155 s 124 s 195 s

if $r_k > \eta$

$x_{k+1} = x_k + d_k$

else

$x_{k+1} = x_k$;

end(for).

We chose the parameters $c = 0.75$, $\eta = 0.01$, $\epsilon = 10^{-8}$ and $\{B_k\}$ was modified by BFGS formula.

If we set $B_k \equiv I$ and $q_k = -g_k$ we can also obtain a new adaptive trust region method denoted by TRI. Zhang’s adaptive trust region method [27] is denoted by TRZ. We chose 18 test problems and their initial points from the literature [13]. For example, P5 means the No. 5 problem and so on. The stop criteria is

$$||g_k|| \leq \epsilon = 10^{-8}.$$ 

We used Visual C++ Language to design the program in a portable computer with Pentium IV/ 1.2MHz CUP. Numerical results are reported in Tables 1–4.
In Table 1, “P” denotes the test problem and n denotes the dimension of problems. Each group of three numbers means the iteration number, function evaluations and gradient evaluations in sequence. “T” refers to the total CPU time for solving all the 18 test problems. As we can see, it is difficult to choose an adequate upper bound $\Delta$ in the original trust region method. If $\Delta$ is too large then the number of solving subproblems will be increased. If $\Delta$ is too small then the efficiency of algorithm will be reduced. Thereby, we should choose an adequate initial trust region radius at each iteration. Such problem does not exist in adaptive trust region methods because the initial trust region radius in adaptive trust region method can be adjusted automatically according to the information of iterates.

However, TRO has an advantage that allows $\eta = 0$. In adaptive trust region method, $\eta \in (0, 1)$. If $\eta = 0$, we don’t know whether the adaptive trust region method can converge. If we take $\Delta = 1, 5, 10, 20, 30$, $\Delta_0 = 0.8$ and $\eta = 0$ in TRO, we have the results in Table 2. TRO(1), TRO(5), TRO(10), TRO(20), TRO(30) denotes the corresponding TRO with $\Delta = 1, 5, 10, 20, 30$, respectively. In Table 2, we only list the total CPU time for solving all the 18 problems.

Similarly, we use TRO(m) to denote TRO with $\Delta = m$, and the related numerical results are listed in Tables 3 and 4.

Tables 2–4 show that TRO is more efficient when we choose $\Delta \in (0.5, 1.5)$ and $\eta \in [0, 0.15]$. This is only a guess. In summary, $\Delta$ in TRO is difficult to determine in practical computation. Adaptive trust region method can overcome this difficulty.

It is shown from Table 1 that TRN seems the best adaptive trust region method because it uses the least total CPU time for solving all the 18 test problems. TRS is the second adaptive trust region method that has good numerical performance. The preliminary numerical results show that the adaptive trust region method is a promising method for optimization problems.

### Table 2. Total CPU time ($\Delta_0 = 0.8$ and $\eta = 0$).

<table>
<thead>
<tr>
<th>TRO</th>
<th>TR0(1)</th>
<th>TRO(5)</th>
<th>TRO(10)</th>
<th>TRO(20)</th>
<th>TRO(30)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>162 s</td>
<td>154 s</td>
<td>178 s</td>
<td>169 s</td>
<td>182 s</td>
</tr>
</tbody>
</table>

### Table 3. Total CPU time ($\Delta_0 = 10$ and $\eta = 0.15$).

<table>
<thead>
<tr>
<th>TRO</th>
<th>TR0(50)</th>
<th>TRO(100)</th>
<th>TRO(150)</th>
<th>TRO(200)</th>
<th>TRO(300)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>198 s</td>
<td>204 s</td>
<td>212 s</td>
<td>185 s</td>
<td>195 s</td>
</tr>
</tbody>
</table>

### Table 4. Total CPU time ($\Delta_0 = 0.2$ and $\eta = 0.01$).

<table>
<thead>
<tr>
<th>TRO</th>
<th>TR0(0.3)</th>
<th>TRO(0.5)</th>
<th>TRO(0.8)</th>
<th>TRO(0.85)</th>
<th>TRO(0.9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>161 s</td>
<td>158 s</td>
<td>168 s</td>
<td>170 s</td>
<td>163 s</td>
</tr>
</tbody>
</table>
Table 5. Iterations, function and gradient evaluations and CPU time.

<table>
<thead>
<tr>
<th>P</th>
<th>n</th>
<th>TRS</th>
<th>TRN</th>
<th>TRI</th>
<th>TRZ</th>
<th>TRO</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>11</td>
<td>32/74/32</td>
<td>26/48/36</td>
<td>31/79/45</td>
<td>35/83/35</td>
<td>58/184/64</td>
</tr>
<tr>
<td>20</td>
<td>31</td>
<td>86/132/79</td>
<td>42/97/63</td>
<td>65/99/64</td>
<td>63/84/63</td>
<td>56/179/56</td>
</tr>
<tr>
<td>21</td>
<td>20</td>
<td>35/69/35</td>
<td>42/64/42</td>
<td>75/147/75</td>
<td>46/53/46</td>
<td>78/126/78</td>
</tr>
<tr>
<td>22</td>
<td>30</td>
<td>54/137/85</td>
<td>38/94/45</td>
<td>69/72/69</td>
<td>53/84/68</td>
<td>59/196/83</td>
</tr>
<tr>
<td>23</td>
<td>20</td>
<td>53/64/56</td>
<td>35/47/46</td>
<td>67/84/67</td>
<td>62/68/62</td>
<td>74/198/78</td>
</tr>
<tr>
<td>24</td>
<td>20</td>
<td>47/52/68</td>
<td>45/58/45</td>
<td>57/83/68</td>
<td>45/83/63</td>
<td>65/174/79</td>
</tr>
<tr>
<td>25</td>
<td>40</td>
<td>64/93/89</td>
<td>62/78/62</td>
<td>78/124/98</td>
<td>78/156/79</td>
<td>63/187/93</td>
</tr>
</tbody>
</table>

CPU – 104 s 79 s 126 s 158 s 174 s

As we can see, the first 18 problems are all small problems. Thus, seven large problems, Problems 19–25, from [13] were used to test the adaptive trust region methods. We take the parameters $\Delta = 1$, $\eta = 0.15$, $c = 0.75$, $\epsilon = 10^{-8}$. Numerical results are listed in Table 5, indicating that TRN is the best adaptive trust region method. We guess that TRN essentially reduces to quasi-Newton method in many situations. Moreover, memory use and matrix computation may play a role in performance comparison beyond the iterative number, function and gradient evaluations. Therefore, CPU time seems a reasonable metric for comparing the numerical performance of algorithms.

6. CONCLUSIONS AND FUTURE RESEARCH

In this paper, we presented a new class of adaptive trust region methods for unconstrained optimization problems and investigated their global convergence. In these new methods, the trust region radius can be adjusted automatically according to the current iterative information by a simple computational formula. Different choices of $q_k$ determine different adaptive trust region methods. If we take $q_k = -g_k$ then the new method will reduce to the recently proposed method [23]. If we take $q_k = -B_k^{-1}g_k$ with $B_k^{-1}$ being available, then we can obtain some interesting new methods that have convergence properties of quasi-Newton method. We can choose other $q_k$ to obtain many different adaptive trust region methods. Numerical results showed that some adaptive trust region methods were available and efficient in practical computation.

For the future research, we should choose $\hat{B}_k$ and $q_k$ by using other approaches and conduct some numerical experiments when $q_k$ is taken as another special direction that satisfies (7).
Acknowledgements. The authors would like to thank two anonymous referees and the editor for their careful reading and valuable comments and suggestions that greatly improved the paper. The first author is very grateful to Professor Y. Yuan for his meaningful guidance when he was as a postdoctoral fellow in Chinese Academy of Sciences.

References


