Continuous Optimization

Nonmonotone adaptive trust region method

Zhenjun Shi\textsuperscript{a,}\textsuperscript{,}, Shengquan Wang\textsuperscript{b}

\textsuperscript{a}Department of Mathematics & Computer Science, Central State University, Wilberforce, OH 45384-1004, USA
\textsuperscript{b}Department of Computer & Information Science, University of Michigan, Dearborn, MI 48128, USA

\section*{1. Introduction}

Consider an unconstrained minimization problem
\begin{equation}
\min f(x), \; x \in \mathbb{R}^n,
\end{equation}
where $\mathbb{R}^n$ denotes an $n$-dimensional Euclidean space and $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function.

Traditional iterative methods for solving (1) are either line search method (LS) or trust region method (TR). LS is based on searching a new iterate along a descent direction at each iteration while TR is based on finding a new iterate within a ball centered at the current iterate at each iteration.

**LS** Generally, LS takes the form
\begin{equation}
x_{k+1} = x_k + \alpha_k d_k, \quad k = 0, 1, 2, \ldots,
\end{equation}
where $d_k$ is a descent direction of $f(x)$ at $x_k$ and $\alpha_k$ is a step size. We expect that the sequence $\{x_k\}$ generated by (2) converges to a solution $x^*$ of (1). For convenience, throughout this paper we denote $F(x) = f(x), \; G(x) = \nabla f(x), \; G_k = \nabla f(x_k)$, and $f(x) = f(x^*)$.

In LS, we can choose different search direction: $d_k = -G_k^{-1}g_k$ (if $G_k$ exists and is invertible) leads to the Newton method (e.g. [1,24]) while $d_k = -g_k$ results in the steepest descent method (e.g. [17,19]). The search direction $d_k$ is generally required to satisfy
\begin{equation}
g_k^T d_k < 0,
\end{equation}
which guarantees that $d_k$ is a descent direction of $f(x)$ at $x_k$ (e.g. [17,38–40]). In order to guarantee the global convergence, we sometimes require $d_k$ to satisfy the sufficient descent condition
\begin{equation}
g_k^T d_k \leq -c\|g_k\|^2,
\end{equation}
where $c > 0$ is a constant. To prove the global convergence of related line search methods, instead of (4), the following angle condition is often used
\begin{equation}
\cos(-g_k, d_k) = -\frac{g_k^T d_k}{\|g_k\| \cdot \|d_k\|} \geq \tau,
\end{equation}
where $(-g_k, d_k)$ represents the angle between $-g_k$ and $d_k$, and $0 < \tau \leq 1$.

After $d_k$ is determined we need to seek a step size along the descent direction and complete one iteration. There are many approaches to find an available step size. Because the exact line search (for step size) is time-consuming and is difficult to implement, the inexact line search is generally used, such as Armijo line search, Goldstein line search, and Wolfe line search [2,17,21,22]. Convergence analysis on line search methods can be found in the literature (e.g. [30,33,34]).

**TR** Unlike LS, TR has the following iteration form (see [26,39,41])
\begin{equation}
x_{k+1} = \begin{cases} x_k + \mu_k d_k, & \text{if } r_k \geq \mu, \\ x_k, & \text{otherwise}, \end{cases}
\end{equation}
for given $\mu \in (0, \frac{1}{3})$, where $r_k$ is defined as
\begin{equation}
r_k = \frac{f_k - f(x_k + \mu_k d_k)}{m_k(0) - m_k(p_k)},
\end{equation}
where $m_k$ is an approximation of $m(x, \mu)$.
and \( p_k \) is a solution to the following trust region subproblem
\[
\min \left\{ m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p : ||p|| \leq \Delta_k \right\}, \tag{8}
\]
where \( B_k \) is an approximation to \( G_k \) and \( \Delta_k \) is a trust region radius.

In (7), \( r_k \) is the ratio of the actual reduction to predicted reduction. When \( r_k \) is close to 1, then the trust region problem model better matches the original problem at the current iterate \( x_k \). In (8), \( m_k(p) \) is a quadratic function of \( p \) that approximates to \( f(x_k + p) \) on the trust region. When \( p = p_k \) is the solution of (8), \( m_k(p) \) will be the optimal approximation to \( f(x_k + p) \).

If \( ||p|| \) in (8) is the Euclidean norm, then the solution \( p_k \) of (8) is the minimizer of \( m_k(p) \) in the ball with the radius \( \Delta_k \). Thus, the trust region method requires us to solve a sequence of subproblems (8) in which the objective function and constraint (which can be written as \( p^T p \leq \Delta_k^2 \)) are both quadratic. The following algorithm describes the process.

**Algorithm 1.1 (Trust Region).** Given \( \Xi > 0 \), \( \Delta_0 \in (0, \Xi) \), and \( \mu \in [0, \frac{1}{2}] \);

For \( k = 0, 1, 2, \ldots \)
- Obtain \( p_k \) by (approximately) solving (8);
- Evaluate \( r_k \) from (7);
- if \( r_k < \frac{1}{\mu} \)
  - \( \Delta_{k+1} = \frac{1}{2} ||p_k|| \);
- else
  - if \( r_k > 1 - \frac{1}{\mu} \) and \( ||p_k|| = \Delta_k \)
    - \( \Delta_{k+1} = \min(2\Delta_k, \Xi) \);
  - else
    - \( \Delta_{k+1} = \Delta_k \);
- \( x_{k+1} = x_k + p_k \);
- end (for).

Sometimes, we need not to solve (8) exactly, and we may find \( p = p_k \) satisfying
\[
m_k(0) - m_k(p) \geq c_1 ||g_k|| \min \left\{ \Delta_k, \frac{||g_k||}{||B_k||^{1/2}} \right\}, \tag{9}\]
and
\[
||p|| \leq \gamma \Delta_k \tag{10}
\]
for \( \gamma > 1 \) and \( c_1 \in (0, 1] \). Indeed, the exact solution \( p_k \) of (8) satisfies (9) and (10) (\cite{17}).

**Lemma 1.1** \cite{17}. Let \( \mu = 0 \) in Algorithm 1.1. Suppose that \( ||B_0|| \leq \beta_1 \) for some constant \( \beta_1 > 0 \), that \( f \) is continuously differentiable and bounded below on the level set
\[
L_0 = \{ x \in \mathbb{R}^n \mid f(x) \leq f(x_0) \},
\]
and that all approximate solutions of (8) satisfy inequalities (9) and (10) for positive constants \( c_1 \) and \( \gamma \). We then have
\[
\lim_{k \to \infty} ||g_k|| = 0. \tag{11}
\]

**Lemma 1.2** \cite{17}. Let \( \mu \in (0, \frac{1}{2}] \) in Algorithm 1.1. Suppose that \( ||B_k|| \leq \beta_1 \) for some constant \( \beta_1 > 0 \), that \( f \) is Lipschitz continuously differentiable and bounded below on the level set \( L_0 \) and that all approximate solutions of (8) satisfy inequalities (9) and (10) for some positive constants \( c_1 \) and \( \gamma \). We then have
\[
\lim_{k \to \infty} ||g_k|| = 0. \tag{12}
\]

In TR as shown in Algorithm 1.1, the difficulty is how to adjust the trust region radius \( \Delta_k \). If \( \Delta_k \) is very large, then the number of solving subproblems will increase; If \( \Delta_k \) is very small, then the efficiency of algorithm will be reduced. The trust region radius could be adjusted by a proportional constant \( \{11, 12, 18, 23, 25, 36, 37\} \). In order to choose an adaptive trust region radius for TR at each iteration, some approaches have been presented in the recent literature (\cite{7, 11, 13, 20, 25, 26, 35, 41, 42}). By combining line search approach into trust region, some new adaptive trust region methods were proposed \( \{3, 10, 18, 27, 29\} \). Nonmonotone trust region methods were also investigated in \( \{5, 6, 8, 14, 15\} \). However, nonmonotone adaptive trust region methods with line search approach has not been studied, which is the focus of this paper.

In this paper, we propose a new nonmonotone adaptive trust region method with line search approach for unconstrained optimization problems. At each iteration, the method needs to choose a descent direction at first and then give an estimation of trust region radius. The method has an adaptive trust region radius and allow the functional value of iterates to increase within finite iterations and finally decrease after such finite iterations. The nonmonotone approach and adaptive trust region radius enable the trust region method to reduce the number of solving trust region subproblems and so that the cost of computation is cut down substantially. The global convergence and convergence rate of this method are analyzed under some mild conditions.

The rest of this paper is organized as follows. In Section 2, we introduce the adaptive nonmonotone trust region method. In Section 3, we analyze the global convergence. Section 4 will focus on the analysis of convergence rate. Numerical results are reported in Section 5. Conclusion remarks are given in Section 6.

2. New adaptive trust region method

In this paper, we have the following assumptions for the objective function \( f(x) \) that will be used in this paper.

(A1) The objective function \( f(x) \) is continuously differentiable and has a lower bound on \( \mathbb{R}^n \);

(A2) The gradient \( g(x) = \nabla f(x) \) of \( f(x) \) is uniformly continuous on an open convex set \( B \) that contains the level set \( L_0 = \{ x \in \mathbb{R}^n \mid f(x) \leq f(x_0) \} \) for given \( x_0 \), or

(A2) The gradient \( g(x) = \nabla f(x) \) of \( f(x) \) is Lipschitz continuous on an open convex set \( B \) that contains the level set \( L_0 \), i.e., there exists \( L \) such that
\[
||g(x) - g(y)|| \leq L||x - y||, \quad \forall x, y \in B.
\]

It is apparent that (A2) implies (A2).

In our new trust region method, we choose a descent direction at first and then give an estimation of trust region radius at each iteration. The new trust region method can be described as follows:

**Algorithm 2.1 (New Trust Region Method).**

**Step 0** Given \( \mu \in (0, 1) \), \( \rho \in (0, 1) \), a nonnegative integer \( M \), and an initial symmetric positive definite matrix \( B_0 \), choose \( x_0 \in \mathbb{R}^n \) and set \( k := 0; \)
Step 1. If \( |g_k| = 0 \) then stop, else go to Step 2;

Step 2. Define \( m(k) \) such that
\[
m(0) = 0, \quad 0 \leq m(k) \leq \min(m(k-1), M).
\]
(13)

Choose a descent direction \( d_k \) of \( f(x) \) at \( x_k \):

Step 2.1. Choose \( B_k \) such that \( i \) is the smallest nonnegative integer that satisfies
\[
d_k^T B_k d_k = \sum_{i=1}^i \|d_k\|^2 > 0.
\]

Step 2.2. Set \( s_k = -\frac{g_k^T d_k}{\alpha_k d_k} \) and \( \alpha_k \) is the largest \( \alpha \) in \( \{s_k, s_k^2 p, s_k^3 p^2, \ldots \} \)
such that
\[
r_k = \max_{i: \mu(g_k, f_k, s_k, p_k)} \frac{f_k(x_k + p_k) - f(x_k + p_k)}{\|p_k\|} \geq \mu.
\]
(14)

where \( p_k \) is the solution of the trust region subproblem
\[
\min \left\{ m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p : \|p\| \leq \bar{\alpha}|d_k| \right\},
\]
(15)

Step 3. Modify \( B_k \) into \( B_{k+1} \) by using quasi-Newton Formula (such as DFP and BFGS formula) as an approximation to \( G_{k+1} = \nabla^2 f(x_{k+1}) \);

Step 4. set \( k := k + 1 \) and go to Step 1.

In the above algorithm, line search approach is used to find the descent direction \( d_k \) that is used in the estimation of the trust region radius as \( \Delta_k = s_k|d_k| \) in Step 2.2. Such radius is adaptive. By (14), the trust region method is also nonmonotone.

The following lemma shows that Algorithm 2.1 is well-defined:

**Lemma 2.1.** If (A1) holds and \( d_k \) satisfies (3), then the new trust region method is well-defined.

**Proof.** By (A1) we have
\[
\lim_{\|p\| \to 0} \frac{\max_{0 \leq \omega \leq m(k)} [f_k(x_k + p_k) - f(x_k + p)]}{m_k(0) - m_k(p)} = \lim_{\|p\| \to 0} \max_{0 \leq \omega \leq m(k)} [f_k(x_k + p_k) - f(x_k + p)] - g_k^T p + \frac{1}{2} p^T B_k p
\]
\[
\geq \lim_{\|p\| \to 0} \frac{f_k(x_k) - f(x_k) - g_k^T p}{p^T B_k p} > 0.
\]

Therefore, there exists an \( \bar{\alpha} > 0 \) such that
\[
\max_{0 \leq \omega \leq m(k)} [f_k(x_k + p_k) - f(x_k + p_k)] > \alpha_k \|p\| \leq \bar{\alpha}|d_k|, \quad \alpha \in [0, \bar{\alpha}],
\]
which implies that the new trust region method is well-defined. \( \Box \)

In the rest of this paper, we will analyze the convergence of the new adaptive TR method.

### 3. Global convergence

In order to show the global convergence, we need the following lemmas.

**Lemma 3.1.** Assume that Algorithm 2.1 generates an infinite sequence \( \{x_k\} \). Then
\[
m_k(0) - m_k(p_k) \geq -\frac{1}{\alpha_k} g_k^T d_k.
\]
**Proof.** Since \( p = \alpha_k d_k \) is a feasible solution to (15), we have
\[
m_k(0) - m_k(p_k) \geq \alpha_k \left[ g_k^T d_k + \frac{1}{2} \alpha_k d_k^T B_k d_k \right] - \alpha_k \left[ -g_k^T d_k + \frac{1}{2} \alpha_k d_k^T B_k d_k \right] = \alpha_k g_k^T d_k.
\]
\[
\geq -\frac{1}{\alpha_k} g_k^T d_k.
\]
The proof is finished. \( \Box \)

**Lemma 3.2.** Assume that (A1) and (A2) hold and \( d_k \) satisfies (3). Algorithm 2.1 generates an infinite sequence \( \{x_k\} \) and \( \{B_k\} \) is a bounded sequence, that is, there is a \( \beta_0 > 0 \) such that \( \|B_k\| < \beta_0 \) \( \forall k \).

Then there exists \( \eta > 0 \) such that
\[
\max_{0 \leq \omega \leq m(k)} [f_k(x_k + p_k) - f(x_k + p_k)] \geq \eta \left( \frac{g_k^T d_k}{\|d_k\|} \right)^2.
\]
(16)

**Proof.** By Lemma 3.1, we have
\[
\max_{0 \leq \omega \leq m(k)} [f_k(x_k + p_k) - f(x_k + p_k)] \geq \frac{\mu}{2 \beta_1} \frac{g_k^T d_k}{\|d_k\|} \geq \frac{\mu}{2 \beta_1} \frac{g_k^T d_k}{\|d_k\|}.
\]

Thus
\[
\max_{0 \leq \omega \leq m(k)} [f_k(x_k + p_k) - f(x_k + p_k)] \geq \frac{\mu}{2 \beta_1} \frac{g_k^T d_k}{\|d_k\|}, \quad k \in K_1.
\]
(18)

In the case of \( k \in K_2 \), we can see that \( \alpha_k \rho \leq s_k \). Suppose that \( p_k \) is an optimal solution of (15) with respect to \( \alpha_k = \alpha_k / \rho \), \( k \in K_2 \). Then,
\[
\frac{m_k(0) - m_k(p_k)}{m_k(0) - m_k(p_k)} < \mu,
\]
i.e.,
\[
\max_{0 \leq \omega \leq m(k)} [f_k(x_k + p_k)] \geq \mu [m_k(0) - m_k(p_k)].
\]

Therefore,
\[
f_k(x_k + p_k) \geq \mu [m_k(0) - m_k(p_k)] = \mu \left[ g_k^T p_k + \frac{1}{2} p_k^T B_k p_k \right].
\]

Thus,
\[
f_k(x_k + p_k) - f_k > \mu \left[ g_k^T p_k + \frac{1}{2} p_k^T B_k p_k \right], \quad \forall k \in K_2.
\]
(19)

By using the mean value theorem on the left-hand side of the above inequality, there exists \( \theta_k \in (0,1) \) such that
\[
g_k(x_k + \theta_k p_k) - g_k(x_k) > \mu \left[ g_k^T p_k + \frac{1}{2} p_k^T B_k p_k \right], \quad \forall k \in K_2.
\]

By noting that \( \|B_k\| < \beta_0 \), we have
\[
g_k(x_k + \theta_k p_k) - g_k(x_k) > \mu \left[ g_k^T p_k + \frac{1}{2} \beta_0 \|p_k\|^2 \right], \quad \forall k \in K_2.
\]
(20)
By (A2), (20), the Cauchy–Schwartz inequality, and Lemma 3.1, we have
\[ L\xi_k^2 \|d_k\|^2 + \frac{1}{2} \mu \beta_0 \xi_k^2 \|d_k\|^2 \geq L\|p_k\|^2 + \frac{1}{2} \mu \beta_0 \|p_k\|^2 \]
\[ \geq g(x_k + \theta_k p_k) - g_k \cdot \|p_k\|^2 + \frac{1}{2} \mu \beta_0 \|p_k\|^2 \]
\[ \geq (1 - \mu) g_k \phi_k^2 p_k \]
\[ = \left(1 - \mu\right) \left( g_k \phi_k^2 + \frac{1}{2} \beta_0 \phi_k^\top B_k \phi_k \right) + \frac{1}{2} \left(1 - \mu\right) \beta_0 \phi_k^2 \|d_k\|^2, \quad k \in K_2. \]

By processing the above inequality, we have
\[ \alpha_k \geq -\frac{1 - \mu}{2L + \beta_0} \frac{g_k^\top d_k}{\|d_k\|^2}, \quad k \in K_2. \tag{21} \]

Thus,
\[ \alpha_k \geq -\frac{\rho(1 - \mu)}{2L + \beta_0} \frac{g_k^\top d_k}{\|d_k\|^2}, \quad k \in K_2. \]

By (14), Lemma 3.1, and (21), we have
\[ \max_{0 \leq j < m} |f_{k, j} - f(x_k + p_k)| \geq \mu |m_k(0) - m_k(p_k)| \geq -\frac{1}{2} \alpha_k g_k^\top d_k \]
\[ \geq \rho(1 - \mu) \frac{g_k^\top d_k}{2L + \beta_0} \quad \text{and} \quad k \in K_2. \tag{22} \]

Therefore,
\[ \max_{0 \leq j < m} |f_{k, j} - f(x_k + p_k)| \geq \rho(1 - \mu) \frac{g_k^\top d_k}{2L + \beta_0} \quad \text{and} \quad k \in K_2. \]

Letting
\[ \eta' = \min \left( \frac{\mu}{2\beta_1}, \frac{\rho(1 - \mu)}{2L + \beta_0} \right), \]
by (18) and (22), we obtain that (16) holds. The proof is completed. \( \square \)

**Lemma 3.3.** Assume that the conditions in Lemma 3.2 hold. Then,
\[ \max_{1 \leq j < m} f(x_{k,j}) \leq \max_{1 \leq j < m} f(x_{k-1,j}) - \eta' \]
\[ \times \min_{0 \leq j < m} \left( \frac{g^\top d_{k,j}}{\|d_{k,j}\|} \right)^2, \tag{23} \]
where \( \eta' \) is defined in Lemma 3.2 and thus
\[ \sum_{l=1}^{\infty} \min_{0 \leq j < m} \left( \frac{g^\top d_{k,j}}{\|d_{k,j}\|} \right)^2 < +\infty. \tag{24} \]

**Proof.** By Lemma 3.2, it suffices to show that the following inequality holds for \( j = 1, 2, \ldots, M \),
\[ f(x_{M,j}) \leq \max_{1 \leq j < M} \left( f(x_{M-1,j}) - \eta' \left( \frac{g^\top d_{M,j}}{\|d_{M,j}\|} \right)^2 \right). \tag{25} \]

Since the new nonmonotone line search condition and Lemma 3.2 imply
\[ f(x_{M,1}) \leq \max_{0 \leq j < m} f(x_{M-1,j}) - \eta' \left( \frac{g^\top d_{M}}{\|d_{M}\|} \right)^2, \tag{26} \]
it follows from this and
\[ m(M) \leq M \]
that (25) holds for \( j = 1 \). Suppose that (19) holds for any \( j : 1 \leq j < M - 1 \). With the descent property of \( d_k \), we obtain
\[ \max_{1 \leq j < M} f(x_{M,j}) \leq \max_{1 \leq j < M} f(x_{M-1,j}). \tag{27} \]

By Algorithm 2.1, the induction hypothesis,\( m(M+j) \leq M \), Lemma 3.2, and (27), we obtain
\[ f(x_{M,j+1}) \leq \max_{0 \leq j < m} f(x_{M,j}) - \eta' \left( \frac{g^\top d_{M,j}}{\|d_{M,j}\|} \right)^2 \leq \max_{1 \leq j < M} \max_{1 \leq j < M} f(x_{M-1,j}) - \eta' \left( \frac{g^\top d_{M-1,j}}{\|d_{M-1,j}\|} \right)^2. \]

Thus, (25) is also true for \( j + 1 \). By induction, (25) holds for \( 1 \leq j < M \). This shows that (23) holds.

Since \( f(x) \) is bounded below by (A1), it follows that
\[ \max_{1 \leq j < M} f(x_{M,j}) > -\infty. \]

By summing (23) over \( l \), we can get
\[ \sum_{l=1}^{\infty} \min_{0 \leq j < m} \left( \frac{g^\top d_{M,l,j}}{\|d_{M,l,j}\|} \right)^2 < +\infty. \]

Therefore, (24) holds. The proof is completed. \( \square \)

**Corollary 3.1.** Assume that the conditions in Lemma 3.2 hold. Then,
\[ \lim_{k \to \infty} f_k \leq \lim_{k \to \infty} f(x_k), \tag{28} \]
where \( l(k) \) satisfies
\[ f(x_k) = \max_{0 \leq j < m} f_{k,j}. \]

**Proof.** Since \( m(k+1) \leq (m(k)+1) \), we have
\[ f(x_{k+1,j}) = \max_{0 \leq j < m} \left( f(x_{k+1,j}) \right) \leq \max_{0 \leq j < m} \left( f(x_{k+1,j}) \right) \]
\[ = \max \{ f(x_{k+j}), f(x_{k}) \} = f(x_k). \]

Thus \( \{ f(x_k) \} \) is a monotone non-increasing sequence. (A1) implies that \( \{ f(x_k) \} \) has a bound from below and thus it has a limit. By (23) in Lemma 3.3, we obtain that (28) holds. \( \square \)
The main global convergence is described as follows:

**Theorem 3.1.** Assume that the conditions in Lemma 3.2 hold. Then

\[
\liminf_{k \to \infty} \left( \frac{\|g_k\|^2}{\|d_k\|^2} \right) = 0. \tag{29}
\]

**Proof.** In fact, if (29) does not hold, then we immediately obtain a contradiction to (24) in Lemma 3.3. This shows that (29) holds. \(\Box\)

With more strict requirement of \(d_k\), we have the following corollary:

**Corollary 3.2.** Assume that the conditions in Lemma 3.2 hold and the direction \(d_k\) satisfies (5). Then,

\[
\liminf_{k \to \infty} \|g_k\| = 0.
\]

**Proof.** It is easy to prove by (5) and Theorem 3.1. The proof is completed. \(\Box\)

Under stronger conditions, we have the following stronger convergence:

**Theorem 3.2.** Assume that the conditions in Lemma 3.2 hold and \(d_k\) satisfies (5), Algorithm 2.1 generates an infinite sequence \(\{x_k\}\) and there exists \(\beta > 0\) such that \(\beta^2 \leq d_k^TB_kd_k, \forall k\). Then,

\[
\lim_{k \to \infty} \|g_k\| = 0. \tag{30}
\]

**Proof.** By (24), we have

\[
\lim_{k \to \infty} \min_{0 \leq j \leq M-1} \left( \frac{\|g_{Mj}\|^2}{\|d_{Mj}\|^2} \right) = 0.
\]

By (5), we have

\[
\lim_{k \to \infty} \min_{0 \leq j \leq M-1} \|g_{Mj}\| = 0.
\]

Letting

\[
\|g_{Mj+\phi}\| = \min_{0 \leq j \leq M-1} \|g_{Mj}\|,
\]

we obtain

\[
\lim_{k \to \infty} \|g_{Mj+\phi}\| = 0. \tag{31}
\]

It follows from (A2), the Cauchy–Schwartz inequality, and \(\beta^2 \leq d_k^TB_kd_k, \forall k\), which implies that \(B_k = B_0\), that

\[
\|g_k\| = \|g_{k-1} + g_k\| \leq \|g_{k-1}\| + \|g_k\| \leq L \alpha_k \|d_k\| + \|g_k\|
\]

\[
\leq \alpha_k \|d_k\| + \|g_k\| \leq \left( 1 + \frac{L}{\beta} \right) \|g_k\|.
\]

By letting \(c_2 = \left( 1 + \frac{L}{\beta} \right)\), we have

\[
\|g_k\| \leq c_2 \|g_k\|.
\]

Therefore, for \(1 \leq i \leq M\),

\[
\|g_{M(i+1)}\| \leq c_2 \|g_{M(i+1)+1}\| \leq \cdots \leq c_2^i \|g_{M(i+\phi)}\|.
\]

By (31), we obtain that (30) holds and the proof is finished. \(\Box\)

4. Convergence rate

4.1. Linear Convergence

In order to analyze the convergence rate, we further assume that.

(A3) The sequence \(x_k \to x^* \) as \(k \to \infty\), \(\nabla^2 f(x^*)\) is a positive definite matrix and \(f(x)\) is twice continuously differentiable on \(N(x^*, \epsilon_0) = \{x ||x-x^*|| < \epsilon_0\}\).

With this assumption, we have the following properties for the \(f(x)\):

**Lemma 4.1.** Assume that (A3) holds. Then there exist \(0 < m' < M\) and \(\epsilon \leq \epsilon_0\) such that

\[
m'|y|^2 \leq y^T \nabla^2 f(x)y \leq M'|y|^2, \quad \forall x, y \in N(x^*, \epsilon); \tag{33}
\]

\[
\frac{1}{2} m'|x-x^*|^2 \leq f(x) - f(x^*) \leq \frac{1}{2} M'|x-x^*|^2, \quad \forall x \in N(x^*, \epsilon); \tag{34}
\]

\[
M'|x-y|^2 \geq (g(x) - g(y))^T (x - y) \geq m'|x-y|^2, \quad \forall x, y \in N(x^*, \epsilon); \tag{35}
\]

and thus

\[
M'|x-x^*|^2 \geq g(x)^T (x - x^*) \geq m'|x-x^*|^2, \quad \forall x \in N(x^*, \epsilon). \tag{36}
\]

By (36) and (35) we can also obtain, from the Cauchy–Schwartz inequality, that

\[
M'|x-x^*| \geq \|g(x)\| \geq M'|x-x^*, \quad \forall x \in N(x^*, \epsilon), \tag{37}
\]

and

\[
\|g(x) - g(y)\| \leq M'|x-y|, \quad \forall x, y \in N(x^*, \epsilon). \tag{38}
\]

By (34) and (37) we can also obtain the following relation

\[
\frac{m'}{2M'^2} \|g(x)\|^2 \leq f(x) - f(x^*) \leq \frac{M'}{2m'^2} \|g(x)\|^2, x \in N(x^*, \epsilon). \tag{39}
\]

Its proof can be found in the literature (e.g. [22]).

The following theorem shows that the iterates generated by the trust region method converge at least R-linearly:

**Theorem 4.1.** Assume that (A3) holds and \(d_k\) satisfies (5) and \(\beta^2 \leq d_k^TB_kd_k, \forall k\). Algorithm 2.1 generates an infinite sequence \(\{x_k\}\). Then \(\{x_k\}\) converges to \(x^*\) at least R-linearly:

**Proof.** If (A3) holds then there exists \(k'\) such that \(x_k \in N(x^*, \epsilon_0), \forall k \geq k' \) and (A1) and (A2) hold if \(x_0 \in N(x^*, \epsilon_0)\). By Lemma 3.2 and (5), we have

\[
\max_{0 \leq j \leq k'} \left| f_k - f_{k+1} \right| \geq \eta^2 \left( - \frac{g_k^T d_k}{d_k^T d_k} \right)^2 \geq \eta^2 \epsilon^2 \|g_k\|^2, \quad k \geq k'. \tag{40}
\]

At first, we obtain from (39) and (32), that

\[
f(x_{k+1}) - f(x^*) \leq b(f(x_k) - f(x^*)), \quad \forall k \geq kr, \tag{41}
\]

where

\[
b = \frac{c_2^2 M^3}{m'} > 1.
\]

For any \(l \geq 0\), let \(\psi(l)\) be any index in \([Ml + 1, M(l + 1)]\) for which

\[
f(x_{\psi(l)}) = \max_{1 \leq j \leq M} f(x_{Mj+1}). \tag{42}
\]

By the definition of \(\psi(l)\), (5), and (23), we can get

\[
f(x_{\psi(l)}) \leq f(x_{(l-1)}) - \epsilon_0 \min_{0 \leq j \leq M-1} \|g_{Mj}\|^2, \tag{43}
\]

where

\[
\epsilon_0 = \min_{x \in \Omega} \min_{0 \leq j \leq M-1} \|g_{Mj}\|^2,
\]

and \(\Omega = \{x ||x-x^*|| < \epsilon_0\}\).
where
\[ c_6 = \tau_0^2 \eta' \]
is a positive constant.

Similarly as in the proof of Theorem 3.1 in (Ref. [5]), we can prove that there exist constants \( c_4 \) and \( c_5 \in (0, 1) \) such that
\[
f(x_k) - f(x^*) \leq c_4 c_5^k [f(x_k) - f(x^*)].
\] (43)

By (34) and (43), we have
\[
\frac{1}{2} m^2 \|x_k - x^*\|^2 \leq f(x_k) - f(x^*) \leq c_4 c_5^k [f(x_k) - f(x^*)],
\]
and thus
\[
\|x_k - x^*\| \leq \sqrt{\frac{2 c_4 c_5^k [f(x_k) - f(x^*)]}{m^2}} (\sqrt{c_5})^k.
\] (44)

Letting
\[
\omega = \sqrt{c_5},
\]
we have
\[
R(x_k) \overset{\text{def}}{=} \lim_{k \to \infty} \|x_k - x^*\|^2 = \lim_{k \to \infty} \left( \sqrt{\frac{2 c_4 c_5^k [f(x_k) - f(x^*)]}{m^2}} \right) \omega = \omega < 1
\]
which shows that \( x_k \) converges to \( x^* \) at least R-linearly. □

4.2. Superlinear convergence

We further assume that,

(A4) \( \{b_k\} \) is a sequence of positive definite matrices and \( \|b_k\| \leq b_0, \forall k \).
Algorithm 2.1 with \( d_k = -B_k^{-1} g_k \) satisfies the following condition
\[
\lim_{k \to \infty} \frac{\|b_k - \nabla^2 f(x_k)\|}{\|d_k\|} = 0.
\] (45)

The following lemma shows that with (A3) and (A4), the new method will reduce to quasi-Newton method:

Lemma 4.2. Assume that (A3) and (A4) hold. Algorithm 2.1 generates an infinite sequence \( \{x_k\} \). Then there exists \( k' \) such that
\[
p_k = d_k, \quad x_k = 1, \quad \forall k \geq k'.
\] (46)

Proof. It is obvious that \( s_k = 1 \) and we assert that \( p = d_k \) is a solution to the subproblem
\[
\min_{p} m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T b_k p, \quad \|p\| \leq \|d_k\|.
\]

By (A3) and (A4), for sufficiently large \( k \), we can find a \( \beta > 0 \) such that
\[
\beta \|d_k\|^2 \leq d_k^T b_k d_k.
\]

By Theorem 3.2 and (A3) we have
\[
\lim_{k \to \infty} x_k = x^*, \quad \lim_{k \to \infty} \|d_k\| = 0,
\] (47)

and thus
\[
\lim_{k \to \infty} (x_k + td_k - x^*) = 0,
\] (48)

where \( t \in [0, 1] \). Assumption (A4) implies that
\[
d_k^T b_k - \nabla^2 f(x_k) d_k = o(\|d_k\|^2).
\] (49)

By the mean value theorem, (A3), (47)–(49), for sufficiently large \( k \), we have
\[
\frac{f(x_k + d_k) - \max_{a_j \in (m_k(x_k - x^*))} f(x_k + d_k)}{m_k(0) - m_k(d_k)} \geq f(x_k + d_k) - f_k
\]
\[
= g_k^T d_k + \int_0^1 (1 - t) d_k^T \nabla^2 f(x_k + td_k) d_k dt
\]
\[
= g_k^T d_k + \frac{1}{2} d_k^T b_k d_k
\]
\[
+ \int_0^1 (1 - t) d_k^T [\nabla^2 f(x_k + td_k) - \nabla^2 f(x_k)] d_k dt
\]
\[
+ \frac{1}{2} d_k^T [\nabla^2 f(x_k) - b_k] d_k
\]
\[
= \left[ g_k^T d_k + \frac{1}{2} d_k^T b_k d_k \right] + o(\|d_k\|^2)
\]
\[
\leq \left[ g_k^T d_k + \frac{1}{2} d_k^T b_k d_k \right] + o(\|d_k\|^2)
\]
\[
= - \mu [m_k(0) - m_k(d_k)]
\]
Thus,
\[
\frac{\max_{a_j \in (m_k(x_k - x^*))} f(x_k + d_k) - f(x_k + d_k)}{m_k(0) - m_k(d_k)} \geq \mu,
\]
which implies that there exists \( k' \) making (46) valid.

Based on Lemma 4.2, we can show that the new trust region method has super-linear convergence rate.

Theorem 4.2. Assume that (A3) and (A4) hold. Algorithm 2.1 generates an infinite sequence \( \{x_k\} \). Then \( x_k \) converges to \( x^* \) superlinearly.

Proof. By Theorem 3.2 and Lemma 4.1, we know that \( x_k \to x^* \). By Lemma 2.2, there exists \( k' \) such that (46) holds and we have
\[
x_{k+1} = x_k + d_k, \quad k \geq k',
\]
where \( d_k = -B_k^{-1} g_k \). By the mean value theorem, Lemma 4.1 and (48), it follows that
\[
g_{k+1} - g_k = \int_0^1 \nabla^2 f(x_k + t(x_{k+1} - x_k))(x_{k+1} - x_k) dt
\]
\[
= \int_0^1 \nabla^2 f(x_k + td_k) d_k dt
\]
\[
= \nabla^2 f(x^*) d_k + \int_0^1 [\nabla^2 f(x_k + td_k) - \nabla^2 f(x^*)] d_k dt
\]
\[
= \nabla^2 f(x^*) d_k + o(\|d_k\|),
\]
and thus,
\[
g_{k+1} = g_k + \nabla^2 f(x^*) d_k + o(\|d_k\|) = -B_k d_k + \nabla^2 f(x^*) d_k + o(\|d_k\|)
\]
\[
= -[B_k - \nabla^2 f(x^*)] d_k + o(\|d_k\|).
\]
By (45) and the above equality, we have
\[
\lim_{k \to \infty} \frac{\|g_{k+1}\|}{\|d_k\|} = 0.
\] (50)
From (37) and (50), it follows that
\[
\frac{\|g_{k+1}\|}{\|d_k\|} \geq \frac{m \|x_{k+1} - x^*\|}{\|d_k\|} = \frac{m \|x_{k+1} - x^*\|}{\|x_{k+1} - x_k\|} \geq \frac{m \|x_{k+1} - x^*\|}{\|x_{k+1} - x_k\| + \|x_k - x^*\|}
\]
\[
= m \frac{\|x_{k+1} - x^*\|}{1 + \|x_{k+1} - x_k\|}
\]
and thus,
\[
\lim_{k \to \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0,
\]
which implies that \( x_k \) converges to \( x^* \) superlinearly. □
4.3. Quadratic convergence

If we take $B_k = \nabla^2 f(x_k)$ in the Algorithm 2.1, then (A4) holds and the new method reduces to Newton method, and based on Theorem 4.2 we have the following result:

**Theorem 4.3.** Assume that (A3) holds, $B_k = \nabla^2 f(x_k)$ for sufficiently large $k$. Algorithm 2.1 generates an infinite sequence $\{x_k\}$. Then $\{x_k\}$ converges to $x^*$ at least superlinearly.

**Proof.** In this case, (A4) holds automatically, and thus the results in Theorem 4.2 hold. □

**Theorem 4.4.** Assume that (A3) holds, $B_k = \nabla^2 f(x_k)$ for sufficiently large $k$. Moreover, there exists a neighborhood $N(x^*, \epsilon) = \{x \in \mathbb{R}^n \mid ||x - x^*|| < \epsilon\}$ of $x^*$ with $\epsilon < \epsilon_0$ such that $\nabla^2 f(x)$ is Lipschitz continuous on $N(x^*, \epsilon)$, i.e., there exists $L(\epsilon)$ such that

$$||\nabla^2 f(x) - \nabla^2 f(y)|| \leq L(\epsilon)||x - y||, \quad \forall x, y \in N(x^*, \epsilon).$$

Algorithm 2.1 generates an infinite sequence $\{x_k\}$. Then $\{x_k\}$ converges to $x^*$ quadratically.

**Proof.** By Theorem 3.2, Lemmas 4.1 and 4.2, it follows that $\{x_k\}$ converges to $x^*$ and there exists $\epsilon > 0$ such that for all $k \geq K$, $x_k \in N(x^*, \epsilon)$, $B_k = \nabla^2 f(x_k)$, and $\epsilon_k = 1$. Let $\epsilon_k = x_k - x^*$. By the mean value theorem we have

$$\epsilon_{k+1} = x_{k+1} - x^* = x_k - x^* + \epsilon_k = \epsilon_k - \nabla^2 f(x_k)^{-1}g_k = \epsilon_k - \nabla^2 f(x_k)^{-1}(g_k - g^*)$$

$$= \epsilon_k - \nabla^2 f(x_k)^{-1}\int_0^1 \nabla^2 f(x_k + t\epsilon_k)\epsilon_k dt$$

$$= \nabla^2 f(x_k)^{-1}[\nabla^2 f(x_k)\epsilon_k - \int_0^1 \nabla^2 f(x_k + t\epsilon_k)\epsilon_k dt]$$

$$= \nabla^2 f(x_k)^{-1}\int_0^1 [\nabla^2 f(x_k) - \nabla^2 f(x_k + t\epsilon_k)]\epsilon_k dt,$$

which and (51) imply that

$$||\epsilon_{k+1}|| \leq \int_0^1 [\nabla^2 f(x_k) - \nabla^2 f(x_k + t\epsilon_k)]dt.$$

Therefore,

$$\lim_{k \to \infty} \frac{||\epsilon_{k+1}||}{||\epsilon_k||} \leq \frac{1}{2} \lim_{k \to \infty} L(\epsilon)||\nabla^2 f(x_k)||^2 = \frac{1}{2} L(\epsilon)||\nabla^2 f(x_k)||^2,$$

which implies that $\{x_k\}$ converges to $x^*$ quadratically. □

4.4. Convergence in monotone case

Assume that $M \equiv 0$. Then, the new method reduces to a monotone adaptive trust region method. We can weaken the assumption and have the following results.

**Theorem 4.5.** Assume that (A1) and (A2) hold and $d_k$ satisfies (3). Algorithm 2.1 generates an infinite sequence $\{x_k\}$ and $\{B_k\}$ is a bounded sequence, that is, there is a $\beta_0 > 0$ such that $||B_k|| \leq \beta_0$, $\forall k$. Then

$$\lim_{k \to \infty} \left(- \frac{g_k^T d_k}{||d_k||}\right) = 0.$$

**Proof.** For the contrary, assume that there exists an infinite subset $K \subseteq \{0, 1, 2, \ldots\}$ and an $\epsilon > 0$ such that

$$\frac{g_k^T d_k}{||d_k||} \geq \epsilon, \quad k \in K.$$

Noting that $||B_k|| \leq \beta_0$, there exists a $\beta_1 > 0$ such that $||B_k|| \leq \beta_1$. Let $K_1 = \{k \in K \mid g_k = b_1 = \beta_0\}$ and $K_2 = \{k \in K \mid g_k < b_1\}$. The proof is divided into two stages.

In the case of $k \in K_1$, by Lemma 3.1, we have

$$f_k - f(x_k + p_k) \geq \frac{1}{2} \frac{1}{\beta_1} g_k^T d_k - \frac{1}{2} \frac{1}{\beta_1} g_k^T B_k d_k \geq \frac{1}{2} \frac{1}{\beta_1} g_k^T d_k.$$

If $K_1$ is an infinite subset of $K$, by (A1), we can deduce a contradiction to (53). Thus $K_1$ must be a finite subset of $K$ and $K_2$ should be an infinite subset of $K$. In the case of $k \in K_2$, by Lemma 3.1 and (53), we have

$$f_k - f(x_k + p_k) \geq \frac{1}{2} \frac{1}{\beta_1} g_k^T d_k = \frac{1}{2} \frac{1}{\beta_1} ||g_k|| \frac{1}{\beta_1} (f_k - f(x_k + p_k)) \leq \frac{1}{2} \frac{1}{\beta_1} ||g_k||$$

Thus $x_k ||d_k|| \to 0$, $k \to K_2$, $k \to +\infty$.

Suppose $p_k'$ is an optimal solution of (15) with respect to $x_k' = x_k / p_k$, $k \in K_2$. Then the following inequality hold:

$$f_k - f(x_k + p_k') \leq \frac{1}{2} \frac{1}{\beta_1} ||g_k||.$$

By (A2), the mean value theorem, Lemma 3.1, and (53), and (54), we have

$$\frac{f_k - f(x_k + p_k')}{m_k(0) - m_k(p_k')} = 1,$$

which contradicts (55) for sufficiently large $k \in K_2$. This contradiction shows that there exists no such an infinite subset $K \subseteq \{0, 1, 2, \ldots\}$ and an $\epsilon > 0$ such that (53) holds. Therefore, (52) holds.

The results on convergence rate will be obtained similarly.

5. Numerical results

In Algorithm 2.1, $d_k$ satisfying (5) has a wide scope. Of course, $d_k = -g_k$ is a natural choice. If we take $d_k = -\alpha_k g_k$, then we can obtain a new adaptive trust region method. Algorithm 1.1 with $\Delta = 100$, $\Delta_0 = 50$ and $\mu = 0.01$ is denoted by TRO ([17], Page 68).
We chose the parameters $\rho = 0.75$, $\mu = 0.01$, $\epsilon = 10^{-8}$ and $[b_k]$ was modified by BFGS formula. In the monotone case ([28]), i.e., $M = 0$, the Algorithm 2.1 with $d_k = -g_k$ and $d_k = -B_k^{-1}g_k$ are denoted by TRS and TRN respectively. Moreover, if we take $M = 5$ and $d_k = -B_k^{-1}g_k$ then Algorithm 2.1 with nonmonotone strategy is denoted by TRNN. Zhang’s adaptive trust region method ([42]) is denoted from the literature ([16]). 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/ 1.2 MHz 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 $T$ in the original trust region method. If $T$ is too large then the number of solving subproblems will be increased. If $T$ 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.

### Table 1

<table>
<thead>
<tr>
<th>P</th>
<th>n</th>
<th>TRS</th>
<th>TRN</th>
<th>TRNN</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>17/32/17</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>15/38/16</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>11/22/13</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>11/18/14</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>18/24/23</td>
<td>37/87/37</td>
<td>28/72/72</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>18/49/18</td>
<td>15/15/15</td>
<td>13/16/13</td>
<td>35/38/35</td>
<td>28/83/78</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>48/62/48</td>
<td>35/35/35</td>
<td>33/33/33</td>
<td>64/68/64</td>
<td>83/75/72</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>49/56/46</td>
<td>37/37/37</td>
<td>37/37/37</td>
<td>66/74/66</td>
<td>85/83/81</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>28/34/28</td>
<td>23/23/23</td>
<td>24/36/24</td>
<td>56/61/56</td>
<td>63/72/64</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>46/62/46</td>
<td>38/38/38</td>
<td>22/27/23</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>35/35/35</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>38/38/38</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>26/26/26</td>
<td>68/78/68</td>
<td>86/92/80</td>
</tr>
<tr>
<td>15</td>
<td>4</td>
<td>76/76/76</td>
<td>59/59/59</td>
<td>44/44/44</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>15/15/15</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>68/68/68</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>28/28/28</td>
<td>54/59/54</td>
<td>63/82/78</td>
</tr>
<tr>
<td>CPU –</td>
<td>118 seconds</td>
<td>59 seconds</td>
<td>53 seconds</td>
<td>124 seconds</td>
<td>195 seconds</td>
<td></td>
</tr>
</tbody>
</table>

### Table 2

<table>
<thead>
<tr>
<th>P</th>
<th>n</th>
<th>TRS</th>
<th>TRN</th>
<th>TRNN</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>21/29/25</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>35/39/36</td>
<td>63/84/63</td>
<td>56/179/56</td>
</tr>
<tr>
<td>21</td>
<td>35/69/35</td>
<td>42/44/62</td>
<td>35/37/35</td>
<td>46/53/46</td>
<td>78/126/78</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>30</td>
<td>54/173/85</td>
<td>38/94/45</td>
<td>25/25/25</td>
<td>53/84/68</td>
<td>59/196/83</td>
</tr>
<tr>
<td>23</td>
<td>30/54/36/</td>
<td>33/47/46</td>
<td>37/37/37</td>
<td>62/68/62</td>
<td>74/198/78</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>40</td>
<td>47/52/68</td>
<td>45/58/45</td>
<td>35/35/35</td>
<td>45/83/63</td>
<td>65/174/79</td>
</tr>
<tr>
<td>CPU –</td>
<td>104 seconds</td>
<td>79 seconds</td>
<td>72 seconds</td>
<td>158 seconds</td>
<td>174 seconds</td>
<td></td>
</tr>
</tbody>
</table>

However, TRO has an advantage that allows $\mu = 0$. In adaptive trust region method, $\mu 
\in (0, 1)$. If $\mu = 0$, we do not know whether the adaptive trust region method can converge. If we take $\mu = 1.5, 10, 20, 30$, $\mu = 0.8$ and $\mu = 0$ in TRO, we have the results in Table 2. TRO(1), TRO(5), TRO(10), TRO(20), TRO(30) denotes the corresponding TRO with $\mu = 1.5, 10, 20, 30$, 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 $\mu = m$, and the related numerical results are listed in Tables 3 and 4. Tables 2-4 show that TRO is more efficient when we choose $\mu \in (0.5, 1.5)$ and $\mu \in [0, 0.15]$. This is only a guess. In summary, $\mu$ in TRO is difficult to determine in practical computation. Adaptive trust region method can overcome this difficulty.

As we can see, the first 18 problems are all small problems. Thus, seven large problems, Problems 19-25, from ([16]) were used to test the adaptive trust region methods. We take the parameters $T = 1, \mu = 0.15, \rho = 0.75, \epsilon = 10^{-8}$. Numerical results are listed in Table 5, indicating that TRNN is the best adaptive trust region method. We guess that TRNN essentially reduces to BFGS 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.

It is shown from Tables 1 and 5 that TRNN seems the best adaptive trust region method because it uses the least total CPU time for solving all the 25 test problems. TRN is the second adaptive trust region method that has good numerical performance. We have observed that the new nonmonotone adaptive trust region method, TRNN, performs better than others, especially in the initial several steps.

Actually, $d_k$ can be any descent direction of the objective function $f(x)$ at the point $x_k$. We are not sure which choice of $d_k$ is the best one. It depends on the status of the objective function $f(x)$ at the point $x_k$. This is a challenging problem. We need to do more numerical research. Preliminary numerical results show that $d_k = -B_k^{-1}g_k$ is a good choice if $B_k^{-1}$ is available. Moreover, it is better to choose a small $M$ in the new method. If $M$ is too big, then the computer needs to memorize more information and do more computation. How to choose $M$ is still a problem. However, $M \in \{2, 3, \ldots, 10\}$ seems a good choice.

The preliminary numerical results show that the adaptive trust region method with nonmonotone strategy is a promising method for optimization problems.
6. Conclusion Remarks

In this paper, we proposed a new nonmonotone adaptive trust region method for unconstrained optimization problems. At each iteration, the method needs to choose a descent direction and then gives an estimation of trust region radius. The adaptive trust region radius and nonmonotone approach allows the functional value of iterates to increase within finite iterations and finally decrease after such finite iterations. This method can reduce the number of solving trust region subproblems substantially because of adaptive trust region subproblem and the nonmonotone approach. The Method can also has a strong convergence under some mild conditions. The global convergence and convergence rate of this method were analyzed.

Actually, we can relax the requirement of $p_k$. We need not to solve (15) exactly, and we may find a $p = p_k$ satisfying

$$m_k(0) - m_k(p) \geq -c_2\gamma_2\frac{g_k}{d_k}, \quad (56)$$

and

$$\|p\| \leq \gamma_1\frac{\gamma_2}{d_k}. \quad (57)$$

for $\gamma \geq 1$ and $c_2 \in (0,1]$. Indeed, the exact solution of (15) satisfies (56) and (57). From the proof of Lemmas 3.1 and 3.2, we can obtain the same convergence conclusion by using (56) and (57).

From the convergence analysis of this trust region method, we can find that this new method is based on the choice of $d_k$, which is commonly used in line search method. Different choices of $d_k$ will lead to different nonmonotone adaptive trust region methods. For example, $d_k = \nabla g_k$ and $d_k = -\nabla g_k$ will result in different trust region methods. This also shows that trust region method and line search method are closely related to each other.

To put it in detail, if we let $p_k = \nabla d_k$ in the proposed nonmonotone adaptive trust region method, then we have the accepted condition

$$r_k = \max_{g_k \in \partial^2 f(x_k)} \{f(x_k) + \langle g_k, p_k \rangle + \frac{1}{2} p_k^T B_k^{-1} p_k \} \geq \mu$$

which sounds like in line search method ([30–32], although $p_k \neq \nabla d_k$ at general situations. This implies that the new nonmonotone adaptive trust region method possesses advantages of trust region method and line search method.

For nonmonotone approach, we should choose $M$ and $m(k)$, in which $m(k)$ can vary at each iteration. Small $M$ is often used in algorithm design. For example, $M = 2, 3, \ldots, 10$.

Acknowledgement

The authors are very grateful to the editor and three reviewers for their valuable comments and suggestions that greatly improved the paper.

References