J Math Model Algor (2015) 14:453–467
DOI 10.1007/s10852-015-9279-y

An Adaptive Trust Region Method Based on Simple Conic Models

Qunyan Zhou[1] · Chun Zhang[2]

Received: 10 July 2014 / Accepted: 9 March 2015 / Published online: 26 March 2015

Abstract A new adaptive trust region algorithm with simple conic models is proposed. By use of the simple conic model, the new method needs less memory capacitance and computational complexity. The nonmonotone and adaptive techniques are introduced to improve the efficiency of the proposed algorithm. The convergence results of the method are proved under certain conditions. Numerical tests show that the new algorithm is efficient and robust.

Keywords Trust region method · Adaptive technique · Simple conic model · Global convergence

Mathematics Subject Classification (2010) 65K05 · 90C30

1 Introduction

Consider the following unconstrained optimization problem

$$\min_{x \in R^n} f(x), \tag{1.1}$$

where $f : R^n \to R$ is continuously differentiable function.

✉ Qunyan Zhou
zhouqunyan@jsut.edu.cn

Chun Zhang
smilingchun@126.com

[1] School of Mathematics and Physics, Jiangsu University of Technology, Changzhou 213001, China

[2] Institute of Science, PLA University of Science and Technology, Nanjing 211101, China

Trust region methods for solving problem (1.1) have very nice global and local convergence properties. A basic trust region algorithm works as follows. At each iterate x_k, it obtains a trial step d_k by solving the following quadratic model subproblem

$$\begin{aligned} &\min\ q_k(d) = f_k + g_k^T d + \frac{1}{2} d^T B_k d, \\ &\text{s.t.}\quad \|d\| \le \Delta_k, \end{aligned} \tag{1.2}$$

where $f_k = f(x_k)$, $g_k = \nabla f(x_k)$, $B_k \in R^{n\times n}$ is a symmetric matrix which is the Hessian approximation of $f(x)$ at the current point x_k, $\Delta_k > 0$ is called the trust region radius and $\|\cdot\|$ refers to the Euclidean norm. The ratio ρ_k between the actual reduction in the function value $f(x_k) - f(x_k + d_k)$ and the predicted reduction $q_k(0) - q_k(d_k)$ plays a key role to decide whether the trial step is acceptable or not and how to adjust the trust region radius.

Conic model methods, a generalization of quadratic model methods, can incorporate more information in the iterations, and provide an effective means for optimization problems [5, 9]. A typical conic model subproblem is

$$\begin{aligned} &\min\ \phi_k(d) = f_k + \frac{g_k^T d}{1 + h_k^T d} + \frac{1}{2}\frac{d^T B_k d}{(1 + h_k^T d)^2}, \\ &\text{s.t.}\quad \|d\| \le \Delta_k, \end{aligned} \tag{1.3}$$

where $h_k \in R^n$ is a horizon vector satisfying $1 + h_k^T d > 0$. If $h_k = 0$, $\phi_k(d)$ is reduced to be quadratic. Also one can see that $\phi_k(d)$ is quadratic along any direction $d \in R^n$ satisfying $h_k^T d = 0$.

The usual trust region methods generate a sequence $\{x_k\}$ such that $\{f(x_k)\}$ is monotonically decreasing. In fact, the methods requiring monotonically decreasing of the objective function values at each iteration may slow the rate of convergence in the presence of a narrow valley. In [6], Grippo et al. proposed a nonmonotone line search technique, in which the stepsize α_k satisfies

$$f(x_k + \alpha_k p_k) \le f_{l(k)} + \gamma \alpha_k g_k^T p_k, \tag{1.4}$$

where $f_{l(k)} = \max\limits_{0\le j\le m_k} \{f_{k-j}\}$, $m_0 = 0, 0 \le m_k \le \min\{m_{k-1} + 1, M\} (k \ge 1)$, $M \ge 0$ is an integer, $0 < \gamma < 1$ and p_k is a descent direction. This nonmonotone technique was generalized to the trust region method in [4, 16]. However, it has some disadvantages. For example, it follows from Eq. 1.4 that a good function value generated at any iteration may be thrown away due to the maximum, and the numerical results are dependent on the choice of parameter M. Zhang and Hager [14] replaced the maximum function value in Eq. 1.4 with an average of function values, that is, they required decreasing of an average of the successive function values. In detail, their method selects α_k satisfying

$$f(x_k + \alpha_k d_k) \le C_k + \gamma \alpha_k g_k^T p_k, \tag{1.5}$$

where

$$C_k = \begin{cases} f_k, & k = 0, \\ \frac{\eta_{k-1} Q_{k-1} C_{k-1} + f_k}{Q_k}, & k \ge 1, \end{cases} \quad Q_k = \begin{cases} 1, & k = 0, \\ \eta_{k-1} Q_{k-1} + 1, & k \ge 1, \end{cases} \tag{1.6}$$

and $\eta_{k-1} \in [\eta_{\min}, \eta_{\max}]$, $\eta_{\min} \in [0, 1)$ and $\eta_{\max} \in [\eta_{\min}, 1)$ are two chosen parameters. Numerical results showed that this nonmonotone technique was superior to Eq. 1.4. Then, this nonmonotone technique was also applied to the trust region methods [10, 13]. Recently,

Gu and Mo [7] found that updating η_k and Q_k at each iteration becomes an encumbrance. To overcome the limitation, Gu and Mo [7] introduced another nonmonotone strategy. They replaced C_k in Eq. 1.5 with R_k which is a simple convex combination of the previous R_{k-1} and f_k, i.e.,

$$R_k = \begin{cases} f_k, & k = 1, \\ \eta_k R_{k-1} + (1 - \eta_k) f_k, & k \geq 2 \end{cases} \tag{1.7}$$

for $\eta_k \in [\eta_{\min}, \eta_{\max}]$. Numerical experiments showed that this nonmonotone technique was efficient and robust.

The choice of the trust region radius is very important to the efficiency of the trust region methods. The adjustment strategy, in which the trust region radius is updated only by simply enlarging or reducing the initial trust region radius at a constant rate, does not make full use of the information at the current iterate point, such as the first-order and second-order derivatives. Hence, many authors have studied the self-adaptive trust region method [5, 15]. In [12], a new efficient self-adaptive adjustment strategy for updating the trust region radius was proposed. That is, at each iteration point k, given $0 \leq \mu_1 < \mu_2 < 1$, $0 < c_1 < 1 < c_2$, set $\Delta_{k+1} = \lambda_{k+1} \|g_{k+1}\| \|B_{k+1}^{-1}\|$, where

$$\lambda_{k+1} = \begin{cases} c_1 \lambda_k, & \text{if } \rho_k < \mu_1; \\ \lambda_k, & \text{if } \mu_1 \leq \rho_k \leq \mu_2; \\ c_2 \lambda_k, & \text{if } \rho_k > \mu_2. \end{cases}$$

Solving problems 1.2 or 1.3 is a key work in the trust region method. Many authors have studied the problem and proposed a lot of methods [4, 11, 17]. Generally, it is costly, especially when B_k is large scale and dense. [18] proposed a simple quadratic trust region subproblem using a scalar approximation of the minimizing function's Hessian. Based on the Taylor's theorem, $\gamma(x_k)I$ is considered as an approximation of B_k in problem (1.2), where $\gamma(x_k)$ is a positive scalar. As a result, the new subproblem could be also resolved easily.

In this paper, we will present a new adaptive simple conic trust region method based on problem (1.3). We use a scalar approximation of the minimizing function's Hessian in the conic trust region subproblem (1.3), and then combine the new trust region method with the nonmonotone technique proposed by Gu and Mo [7] and a new adaptive technique. In the next section, we present the new algorithm in detail. In Section 3, a convenient method to get an inexact solution to the subproblem is presented. The convergence results are discussed in Section 4. Section 5 presents some preliminary numerical results of our new algorithm. Finally, we present some concluding remarks.

2 New Algorithm

Motivated by the idea in [18], we use a scalar matrix $\gamma_k I\,(\gamma_k > 0)$ to approximate the B_k in the conic model. Construct the new subproblem

$$\begin{aligned} \min\ & \phi_k(d) = f_k + \frac{g_k^T d}{1 + h_k^T d} + \frac{1}{2} \frac{\gamma_k d^T d}{(1 + h_k^T d)^2}, \\ \text{s.t.}\ & 1 + h_k^T d > 0, \\ & \|d\| \leq \Delta_k, \end{aligned} \tag{2.1}$$

which is called as the simple conic trust region subproblem. We can see that the inequality constraint $1 + h_k^T d > 0$ enters the subproblem (2.1).

Suppose d_k is the solution of the (2.1), then either $x_k + d_k$ is accepted as a new iteration point or the trust region is reduced according to a comparison between the actual reduction

$$\text{Ared}_k(d_k) = R_k - f(x_k + d_k),$$

and the predicted reduction of the model

$$\text{Pred}_k(d_k) = \phi_k(0) - \phi_k(d_k),$$

i.e.,

$$\rho_k = \frac{\text{Ared}_k(d_k)}{\text{Pred}_k(d_k)} = \frac{R_k - f(x_k + d_k)}{\phi_k(0) - \phi_k(d_k)}, \tag{2.2}$$

where R_k is computed by Eq. 1.7. That is, if $\text{Ared}_k(d_k)$ is satisfactory compared with $\text{Pred}_k(d_k)$, then we finish the current iteration by taking $x_{k+1} = x_k + d_k$ and adjusting the trust region radius; otherwise, the iteration is repeated at point x_k with a reduced trust region radius.

After d_k is accepted, the scalar γ_{k+1} and the horizon vector h_{k+1} can be updated for the next iteration. Consider a conic model of $f(x)$ around x_{k+1},

$$\phi_{k+1}(d) = f_{k+1} + \frac{g_{k+1}^T d}{1 + h_{k+1}^T d} + \frac{1}{2}\frac{\gamma_{k+1} d^T d}{(1 + h_{k+1}^T d)^2}. \tag{2.3}$$

The gradient of $\phi_{k+1}(d)$ is

$$\nabla\phi_{k+1}(d) = \frac{1}{1 + h_{k+1}^T d}\left(I - \frac{h_{k+1} d^T}{1 + h_{k+1}^T d}\right)\left(g_{k+1} + \frac{\gamma_{k+1} d}{1 + h_{k+1}^T d}\right).$$

The conic model (2.3) satisfies

$$\phi_{k+1}(0) = f_{k+1}, \nabla\phi_{k+1}(0) = g_{k+1}, \tag{2.4}$$

$$\phi_{k+1}(-d_k) = f_k, \nabla\phi_{k+1}(-d_k) = g_k. \tag{2.5}$$

Set

$$\beta = 1 - h_{k+1}^T d_k, \tag{2.6}$$

then the two interpolation conditions in Eq. 2.5 lead to

$$2\beta^2(f_k - f_{k+1}) = -2\beta g_{k+1}^T d_k + \gamma_{k+1} d_k^T d_k, \tag{2.7}$$

$$\beta^3 g_k = \left(\beta I + h_{k+1} d_k^T\right)(\beta g_{k+1} - \gamma_{k+1} d_k), \tag{2.8}$$

respectively. Pre-multiply Eq. 2.8 by d_k^T, we can obtain

$$\beta^3 d_k^T g_k = \beta d_k^T g_{k+1} - \gamma_{k+1} d_k^T d_k. \tag{2.9}$$

It follows from Eqs. 2.7 and 2.9 that

$$\beta^2 g_k^T d_k + 2\beta(f_k - f_{k+1}) + g_{k+1}^T d_k = 0 \tag{2.10}$$

which is a quadratic equation with β. Equation 2.10 has real roots if and only if

$$\rho = (f_k - f_{k+1})^2 - \left(g_{k+1}^T d_k\right)\left(g_k^T d_k\right) \geq 0,$$

and we can get the solution

$$\beta = \frac{f_k - f_{k+1} + \sqrt{\rho}}{-g_k^T d_k}.$$

If $\rho < 0$, then set $\beta = 1$. That is

$$\beta = \begin{cases} \frac{f_k - f_{k+1} + \sqrt{\rho}}{-g_k^T d_k}, & \text{if } \rho \geq 0; \\ 1, & \text{otherwise.} \end{cases}$$

As one of the easiest special solution of Eq. 2.6, we take

$$h_{k+1} = \frac{1-\beta}{g_k^T d_k} g_k. \tag{2.11}$$

From Eq. 2.7 we have that γ_{k+1} satisfies

$$\gamma_{k+1} d_k^T d_k = 2\beta^2 (f_k - f_{k+1}) + 2\beta g_{k+1}^T d_k.$$

So,

$$\gamma_{k+1} = \frac{2}{d_k^T d_k}\left[\beta^2 (f_k - f_{k+1}) + \beta g_{k+1}^T d_k\right]. \tag{2.12}$$

By Eq. 2.10, we know that γ_{k+1} defined in Eq. 2.12 can be expressed as

$$\gamma_{k+1} = \frac{1}{d_k^T d_k}\left(\beta g_{k+1}^T d_k - \beta^3 g_k^T d_k\right),$$

which can also be deduced by Eq. 2.9 directly.

In order to make the approximation of the Hessian matrix positive definite, we must consider the situation when $\gamma_{k+1} \leq 0$. Select a small constant $\delta > 0$ and set

$$\theta_{k+1} = \beta^2 \delta - \beta^2 (f_k - f_{k+1}) - \beta g_{k+1}^T d_k,$$

then γ_{k+1} can be modified as follows

$$\gamma_{k+1} = \frac{2}{d_k^T d_k}\left[\beta^2 (f_k - f_{k+1}) + \beta g_{k+1}^T d_k + \theta_{k+1}\right] = \frac{2\beta^2 \delta}{d_k^T d_k},$$

which is obviously positive. If we set

$$\widehat{\gamma}_{k+1} = \frac{2}{d_k^T d_k}\left[\beta^2 (f_k - f_{k+1}) + \beta g_{k+1}^T d_k\right],$$

then

$$\gamma_{k+1} = \begin{cases} \widehat{\gamma}_{k+1}, & \text{if } \widehat{\gamma}_{k+1} > 0; \\ \frac{2\beta^2 \delta}{d_k^T d_k}, & \text{otherwise.} \end{cases} \tag{2.13}$$

Now we give a description of nonmonotone adaptive trust region method based on simple conic models.

Algorithm 1 (NASCTR)

Step 0. *Given* $x_0 \in R^n$, $\Delta_0 > 0$, $\Delta_{\max} > 0, 0 < \mu < \mu_1 < \mu_2 < 1, 0 < c_1 < 1 < c_2, \lambda_0 = 1, \epsilon > 0, \delta > 0, \theta > 0$. *Set* $k = 0$, $\gamma_0 = 1$, $h_0 = \mathbf{0}$. *Choose* $\eta_{\min} \in [0, 1)$ *and* $\eta_{\max} \in [\eta_{\min}, 1)$, $R_0 = f_0$.

Step 1. *If test criteria are fulfilled, then stop. Otherwise, goto Step 2.*

Step 2. *Solve the subproblem (2.1) for* d_k.

Step 3. *Compute* $\mathrm{Ared}_k(d_k)$, $\mathrm{Pred}_k(d_k)$, ρ_k.

Step 4. *If* $\rho_k < \mu$, *set* $\Delta_k = c_1\Delta_k$, *goto Step2.*

Step 5. *Set* $x_{k+1} = x_k + d_k$ *and compute* g_{k+1}.

Step 6. *Update* h_{k+1} by Eq. 2.11 *and compute* γ_{k+1} *according to* Eq. 2.13.

Step 7. *If* $\gamma_{k+1} \le \epsilon$ *or* $\gamma_{k+1} \ge \frac{1}{\epsilon}$, *set* $\gamma_{k+1} = \theta$.

Step 8. *Compute* $\Delta = \lambda_{k+1} \max\{\frac{1}{\gamma_{k+1}}, \frac{1}{\gamma_{k+1} + h_{k+1}^T g_{k+1}}\}\|g_{k+1}\|$, *where*

$$\lambda_{k+1} = \begin{cases} c_1\lambda_k, & \text{if } \rho_k < \mu_1; \\ \lambda_k, & \text{if } \mu_1 \le \rho_k \le \mu_2; \\ c_2\lambda_k, & \text{otherwise.} \end{cases}$$

Set $\Delta_{k+1} = \min\{\Delta, \Delta_{\max}\}$.

Step 9. *Update* R_{k+1} *and set* $k = k + 1$, *goto Step 1.*

Remark 2.1 (1) The procedure of "Step 2-Step 3-Step 4-Step 2" is called as inner cycle.

(2) In order to keep the trust region $\{d | \|d\| \le \Delta_k\}$ in the side of the superplane $\{d \mid 1 + h_k^T d > 0\}$, we assume that $\|h_k\|\Delta_k < 1$ and set $\Delta_k = \alpha/\|h_k\|$ when $\|h_k\|\Delta_k \ge 1$, where $0 < \alpha < 1$.

(3) The object of Step 7 is to avoid uphill directions and to keep the sequence $\{\gamma_k\}$ uniformly bounded. In fact, for all k,

$$0 < \min(\epsilon, \theta) \le \gamma_k \le \max\left(\frac{1}{\epsilon}, \theta\right). \tag{2.14}$$

(4) If $h_k = \mathbf{0}$ for all $k \ge 1$, then Algorithm 1 is reduced to the algorithm in [18]. We denote it as NASQTR.

3 Solution of the Simple Conic Trust Region Subproblem

In this section, we discuss how to solve the subproblem (2.1). If $\gamma_k + h_k^T g_k \ne 0$, then the unique minimizer point of the conic function $\phi_k(d)$ in (2.1) is

$$d_k^N = -\frac{g_k}{\gamma_k + h_k^T g_k}. \tag{3.1}$$

The following theorem is obvious.

Theorem 3.1 *Suppose that* $\gamma_k + h_k^T g_k \ne 0$ *and* d_k^N *is defined as in* Eq. 3.1. *If* $\|d_k^N\| \le \Delta_k$, *then* $d_k = d_k^N$ *is the optimal solution of* problem (2.1); *otherwise the optimal solution* d_k *of* problem (2.1) *will locate at the boundary of the trust region.*

Proof The proof is similar to that of the theorem in [11, 17]. □

The algorithm for solving problem (2.1) approximately can be described as follows.

Algorithm 2

Step 1. *If* $\gamma_k + h_k^T g_k \neq 0$, *compute* d_k^N. If $\|d_k^N\| \leq \Delta_k$, *set* $d_k = d_k^N$ *and return; otherwise, goto Step 2.*

Step 2. *Set* $d_k = -\frac{\Delta_k}{\|g_k\|} g_k$ *and return.*

It is obvious that the approximate solution d_k obtained by Algorithm 2 is feasible to (2.1). In the following, we will prove that the d_k computed by Algorithm 2 can guarantee enough decrease of the conic model.

Lemma 3.2 *Suppose that* h_k *and* Δ_k *are all bounded above, i.e., there exist constants* $\Delta_{max} > 0$ *and* $M_h > 0$ *such that* $\Delta_k \leq \Delta_{max}$ *and* $\|h_k\| \leq M_h$ *for all* k. *Then*

$$\mathrm{Pred}_k(d_k) \geq \psi \|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\gamma_k}\right\}, \tag{3.2}$$

where $\psi = \frac{1}{2(1+\Delta_{max} M_h)}$.

Proof Case 1. $d_k = d_k^N = -\frac{g_k}{\gamma_k + h_k^T g_k}$.

$$\mathrm{Pred}_k(d_k) = \phi_k(0) - \phi_k(d_k) = \frac{g_k^T g_k}{\gamma_k} - \frac{1}{2}\frac{\gamma_k g_k^T g_k}{\gamma_k^2} = \frac{1}{2}\frac{\|g_k\|^2}{\gamma_k}. \tag{3.3}$$

Case 2. $d_k = -\frac{\Delta_k}{\|g_k\|} g_k$.

(a) If $0 \leq \gamma_k + h_k^T g_k < \frac{\|g_k\|}{\Delta_k}$, then $0 < \gamma_k < \frac{\|g_k\|}{\Delta_k} - h_k^T g_k$.

$$\begin{aligned}
\mathrm{Pred}_k(d_k) &= \phi_k(0) - \phi_k(d_k) = \frac{\Delta_k \|g_k\|^2}{\|g_k\| - \Delta_k h_k^T g_k} - \frac{\gamma_k \Delta_k^2 g_k^T g_k}{2(\|g_k\| - \Delta_k h_k^T g_k)^2} \\
&\geq \frac{\Delta_k \|g_k\|^2}{\|g_k\| - \Delta_k h_k^T g_k} - \frac{\Delta_k \|g_k\|^2 (\|g_k\| - \Delta_k h_k^T g_k)}{2(\|g_k\| - \Delta_k h_k^T g_k)^2} \\
&= \frac{\Delta_k \|g_k\|^2}{2(\|g_k\| - \Delta_k h_k^T g_k)} \geq \frac{\Delta_k \|g_k\|^2}{2(\|g_k\| + \Delta_k \|h_k\| \|g_k\|)} \\
&\geq \frac{\Delta_k \|g_k\|}{2(1 + \Delta_{\max} M_h)}.
\end{aligned} \tag{3.4}$$

(b) If $-\frac{\|g_k\|}{\Delta_k} < \gamma_k + h_k^T g_k < 0$, it follows from $\gamma_k > 0$ that $h_k^T g_k < 0$.

$$\begin{aligned}
&\mathrm{Pred}_k(d_k) = \phi_k(0) - \phi_k(d_k^C) = \frac{\Delta_k \|g_k\|^2}{\|g_k\| - \Delta_k h_k^T g_k} - \frac{\gamma_k \Delta_k^2 g_k^T g_k}{2(\|g_k\| - \Delta_k h_k^T g_k)^2} \\
&= \frac{2\Delta_k \|g_k\|^2 (\|g_k\| - \Delta_k h_k^T g_k) - \gamma_k \Delta_k^2 g_k^T g_k}{2(\|g_k\| - \Delta_k h_k^T g_k)^2} \\
&= \frac{\Delta_k \|g_k\|^2 \|g_k\| + \Delta_k \|g_k\|^2 (\|g_k\| - \Delta_k h_k^T g_k) - \Delta_k^2 (\|g_k\|^2 h_k^T g_k + \gamma_k g_k^T g_k)}{2(\|g_k\| - \Delta_k h_k^T g_k)^2} \\
&\geq \frac{\Delta_k \|g_k\|^2}{2(\|g_k\| - \Delta_k h_k^T g_k)} \geq \frac{\Delta_k \|g_k\|}{2(1 + \Delta_{\max} M_h)}.
\end{aligned} \tag{3.5}$$

It follows from Eqs. 3.3, 3.4 and 3.5 that

$$\mathrm{Pred}_k(d_k) \geq \|g_k\| \min\left\{\frac{\|g_k\|}{2\gamma_k}, \frac{\Delta_k}{2(1+\Delta_{\max} M_h)}\right\} \geq \psi \|g_k\| \min\left\{\frac{\|g_k\|}{\gamma_k}, \Delta_k\right\} \tag{3.6}$$

where $\psi = \min\left\{\frac{1}{2}, \frac{1}{2(1+\Delta_{\max} M_h)}\right\} = \frac{1}{2(1+\Delta_{\max} M_h)}$. □

4 Convergence Analysis

In this section, the global convergence of Algorithm 1 is established under the following reasonable assumptions.

Assumption 4.1 (i) The level set $L(x_0) = \{x \in R^n | f(x) \leq f(x_0)\}$ is bounded and $f(x)$ is twice continuously differentiable in $L(x_0)$.

(ii) There exists $M_h > 0$ such that $\|h_k\| \leq M_h$ for all k.

Lemma 4.2 *Let $\{x_k\}$ be the sequence generated by Algorithm 1, then we have*

$$f_{k+1} \leq R_{k+1} \leq R_k, \ \forall k = 1, 2, \cdots. \tag{4.1}$$

Proof By $\rho_k \geq \mu$, Eq. 2.2 and Lemma 3.2, we have for $\forall k$,

$$\begin{aligned} R_k - f_{k+1} &\geq \mu(\phi_k(0) - \phi_k(d_k)) = \mu \mathrm{Pred}_k(d_k) \\ &\geq \mu\psi\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\gamma_k}\right\} > 0, \end{aligned} \tag{4.2}$$

from which we have

$$R_k \geq f_{k+1}, \ \text{for}\, \forall k. \tag{4.3}$$

By the definition of R_k and Eq. 4.3, we can obtain that

$$R_k = \eta_k R_{k-1} + (1-\eta_k) f_k \geq \eta_k f_k + (1-\eta_k) f_k = f_k, \ \text{for}\, k \geq 1$$

and $R_0 = f_0$. So

$$R_k \geq f_k, \ \text{for}\, \forall k. \tag{4.4}$$

On the other hand, from Eq. 4.3, we have

$$R_{k+1} = \eta_{k+1} R_k + (1-\eta_{k+1}) f_{k+1} \leq \eta_{k+1} R_k + (1-\eta_{k+1}) R_k = R_k, \forall k. \tag{4.5}$$

It follows from Eqs. 4.4 and 4.5 that Eq. 4.1 holds. □

Lemma 4.3 *Suppose that Assumption 4.1 holds, the sequence $\{x_k\}$ generated by Algorithm 1 is contained in the level set $L(x_0)$.*

Proof It follows from Lemma 4.2, Assumption 4.1 and $R_0 = f_0$. □

The following lemma guarantees that Algorithm 1 does not cycle infinitely in the inner cycle.

Lemma 4.4 *Suppose that Assumption 4.1 holds, then the inner cycle in Algorithm 1 is well defined.*

Proof Suppose that Algorithm 1 cycles infinitely between Step 2 and Step 4 at iteration k. We define the cycling index at iteration k by $k(i)$, then we have

$$\rho_{k(i)} < \mu, i = 1, 2, \cdots \tag{4.6}$$

and $\Delta_{k(i)} \to 0$ as $i \to \infty$. Besides, when $\Delta_{k(i)} \to 0$, we have $d_{k(i)} \to 0$.

When $\|d_k\|$ is sufficiently close to zero, since $\{\|h_k\|\}$ is bounded, we have $1/(1+h_k^T d_k) = 1 + O(\|d_k\|)$. By the boundedness of $\|g_k\|$ and γ_k, we have

$$\frac{g_k^T d_k}{1 + h_k^T d_k} = g_k^T d_k + O\left(\|d_k\|^2\right), \quad \frac{\gamma_k d_k^T d_k}{(1 + h_k^T d_k)^2} = \gamma_k d_k^T d_k + o\left(\|d_k\|^2\right). \tag{4.7}$$

Since $f(x)$ is twice continuously differentiable in $L(x_0)$, then there exists $K_H > 0$ such that $\|\nabla^2 f(x)\| \le K_H$ holds for all $x \in L(x_0)$.

$$\begin{aligned} &|f_k - f(x_k + d_k) + \frac{g_k^T d_k}{1 + h_k^T d_k} + \frac{1}{2}\frac{\gamma_k d_k^T d_k}{(1 + h_k^T d_k)^2}| \\ &= |-g_k^T d_k - \frac{1}{2} d_k^T \nabla^2 f(x_k + \theta_k d_k) d_k + g_k^T d_k + \frac{1}{2}\gamma_k d_k^T d_k + O\left(\|d_k\|^2\right)| \\ &\le \frac{1}{2}\left(M_H + \max\left(\frac{1}{\epsilon}, \theta\right)\right)\|d_k\|^2 + O\left(\|d_k\|^2\right) = O\left(\|d_k\|^2\right). \end{aligned} \tag{4.8}$$

Then by Eq. 4.8 and Lemma 3.2, we have

$$\begin{aligned} |\frac{f_k - f(x_k + d_{k(i)})}{\text{Pred}_k(d_{k(i)})} - 1| &= \frac{|f_k - f(x_k + d_{k(i)}) + \frac{g_k^T d_{k(i)}}{1+h_k^T d_{k(i)}} + \frac{1}{2}\frac{\gamma_k d_{k(i)}^T d_{k(i)}}{(1+h_k^T d_{k(i)})^2}|}{|\text{Pred}_k(d_{k(i)})|} \\ &\le \frac{O(\|d_{k(i)}\|^2)}{\psi\|g_k\| \min\left\{\Delta_{k(i)}, \frac{\|g_k\|}{\gamma_k}\right\}}. \end{aligned} \tag{4.9}$$

Since $\|d_{k(i)}\| \to 0$ as $i \to \infty$, then from Eq. 4.9 we have

$$\lim_{i\to\infty} \frac{f_k - f(x_k + d_{k(i)})}{\text{Pred}_k(d_{k(i)})} = 1. \tag{4.10}$$

By Eq. 2.2 and Lemma 4.2 we have

$$\rho_{k(i)} = \frac{R_k - f(x_k + d_{k(i)})}{\text{Pred}_k(d_{k(i)})} \ge \frac{f_k - f(x_k + d_{k(i)})}{\text{Pred}_k(d_{k(i)})}. \tag{4.11}$$

From Eqs. 4.10 and 4.11 we know that $\rho_{k(i)} \ge \mu$ for sufficiently large i. This is a contradiction to Eq. 4.6. □

Theorem 4.5 *Suppose that Assumption 4.1 holds. Let $\{x_k\}$ be the sequence generated by Algorithm 1, then we have*

$$\liminf_{k\to\infty} \|g_k\| = 0. \tag{4.12}$$

Proof We will show that $\|g_k\|$ is not bounded away from zero by contradiction. Suppose that there exists a constant τ such that

$$\|g_k\| \ge \tau, \text{ for } \forall k. \tag{4.13}$$

It follows Eq. 2.2 and Lemma 3.2 that

$$f_{k+1} \le R_k - \mu \text{Pred}_k(d_k) \le R_k - \mu\psi\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\gamma_k}\right\}. \tag{4.14}$$

From the definition of R_k, Eqs. 4.13, 4.14 and 2.14 we can obtain that

$$\begin{aligned} R_{k+1} &= \eta_{k+1} R_k + (1-\eta_{k+1}) f_{k+1} \\ &\le \eta_{k+1} R_k + (1-\eta_{k+1}) \left(R_k - \mu\psi \|g_k\| \min\left\{ \Delta_k, \frac{\|g_k\|}{\gamma_k} \right\} \right) \\ &= R_k - (1-\eta_{k+1}) \mu\psi \|g_k\| \min\left\{ \Delta_k, \frac{\|g_k\|}{\gamma_k} \right\}, \\ &\le R_k - (1-\eta_{k+1}) \mu\psi \|g_k\| \min\left\{ \Delta_k, \frac{\tau}{\max\left(\frac{1}{\epsilon}, \theta\right)} \right\}, \end{aligned} \tag{4.15}$$

which implies

$$R_k - R_{k+1} \ge (1-\eta_{k+1}) \mu\psi \|g_k\| \min\left\{ \Delta_k, \frac{\tau}{\max\left(\frac{1}{\epsilon}, \theta\right)} \right\}. \tag{4.16}$$

By Assumption 4.1 we have that $\{f_k\}$ is bounded below in $L(x_0)$. From Lemma 4.2 and Lemma 4.3, we know that $f_k \le R_k$ for all $k \ge 0$ and $\{R_k\}$ is nonincreasing, so $\{R_k\}$ is convergent. It follows from Eq. 4.16 that

$$\lim_{k\to\infty} \min\left\{ \Delta_k, \frac{\tau}{\max\left(\frac{1}{\epsilon}, \theta\right)} \right\} = 0, \tag{4.17}$$

which implies that

$$\lim_{k\to\infty} \Delta_k = 0. \tag{4.18}$$

It follows from the proof of Lemma 4.4 and Eq. 4.18 that $\rho_k \ge \mu_2$ for k large enough. By the description of Algorithm 1, it implies that there exists a positive constant λ^* such that

$$\lambda_k \ge \lambda^* \tag{4.19}$$

for all sufficiently large k. On the other hand,

$$\lim_{k\to\infty} \lambda_{k+1} \max\left\{ \frac{1}{\gamma_{k+1}}, \frac{1}{\gamma_{k+1} + h_{k+1}^T g_{k+1}} \right\} \|g_{k+1}\| = 0. \tag{4.20}$$

By Eqs. 2.14 and 4.13, we have

$$\max\left\{ \frac{1}{\gamma_{k+1}}, \frac{1}{\gamma_{k+1} + h_{k+1}^T g_{k+1}} \right\} \|g_{k+1}\| \ge \frac{\|g_{k+1}\|}{\gamma_{k+1}} \ge \frac{\tau}{\max\left(\frac{1}{\epsilon}, \theta\right)}.$$

It follow form Eq. 4.20 that $\lim_{k\to\infty} \lambda_{k+1} = 0$, which contradicts (4.19). □

Theorem 4.6 *Suppose that Assumption 4.1 holds and $\{x_k\}$ converges to x^* where $\nabla^2 f(x^*)$ is positive definite. If $\lim_{k\to\infty} \frac{\|(\gamma_k I - \nabla^2 f(x^*))d_k\|}{\|d_k\|} = 0$, then the sequence $\{x_k\}$ converges to x^* superlinearly.*

Proof See [11]. □

5 Numerical Results

To examine the precision for the approximation of the original function by our simple conic model, we compare Algorithm 1 without the adaptive technique (NSCTR) with the corresponding algorithm in which B_k is dense and updated by BFGS formula (NBCTR). In NBCTR, the conic model subproblem is solved by the modified dogleg path method[5, 17]. The programs are written in MATLAB. The test problems are selected from [1]. For each test function we have considered the situation with the number of variables $n = 10$.

The parameters are as follows: $\Delta_0 = \|g_0\|, \mu = 0.1, \mu_1 = 0.25, \mu_2 = 0.75, c_1 = 0.5, c_2 = 1.5, \epsilon = 10^{-6}, \alpha = 1 - 10^{-5}, \delta = 0.0001, \eta_k = 0.99, \Delta_{\max} = 100$. While in Step 7, if $\gamma_k \leq \epsilon$, the parameter $\theta = \epsilon$; if $\gamma_k \geq \frac{1}{\epsilon}, \theta = \frac{1}{\epsilon}$. The termination rule is

$$\|g_k\|_\infty \leq 10^{-5}(1 + |f(x_k)|).$$

The numerical results are listed in Table 1. This table contains the name of the problem (Problem), the number of function evaluations (Func), the number of iterations (Iter), the

Table 1 Numerical results of NBCTR and NSCTR

Problem	NBCTR Func/Iter/Fval/Time	NSCTR Func/Iter/Fval/Time
Variably dimensioned	30/9/0.00000/0.0099	30/9/0.00000/0.0051
Penalty I	——	97/87/0.00007/0.0124
Ex. trigonometric	21/21/0.00003/0.0091	52/52/0.00003/0.0088
Ex. rosenbrock	15/12/ 0.00000/0.0059	17/14/0.00000/0.0032
Convex1	4/4/10.00000/0.0370	4/4/10.00000/0.0014
Convex2	18/18/5.50000/0.0069	16/16/5.50000/0.0032
Almost line	20/12/0.00000/0.0066	34/22/0.00000/0.0052
Broyden tridiagnal	5498/3452/244.92127/1.4189	57/35/244.92127/0.0095
Dixon	209/140/0.00000/0.0426	125/119/0.00000/0.0161
Perturbed quadratic	34/25/0.00000/0.0092	30/26/0.00000/0.0044
Ex. tridigonal-1	28/19/0.00000/0.0082	11/9/0.00000/0.0041
Ex. Beal	17/12/0.00000/0.0059	49/39/0.00000/0.0065
Digonal 1	34/27/-47.08283/0.0095	22/20/-47.08283/0.0039
Digonal 2	20/20/5.62115/0.0067	16/16/5.62115/0.0036
Digonal 3	30/23/-21.20431/0.0095	22/19/-21.20431/0.0034
Digonal 4	11/5/0.00000/0.0047	14/8/0.00000/0.0024
Digonal 5	5/5/6.93147/0.0034	5/5/6.93147/0.0030
Quadrtic diagonal perturbed	19/15/0.00000/0.0068	17/13/0.00000/0.0033
Almost perturbed quadratic	34/25/0.00000/0.0099	31/27/0.00000/0.0045
Hager	18/18/3.19506/0.0065	17/13/3.19506/0.0029
Generalized PSCI	114/108/9.00000/0.0263	17/11/9.00000/0.0030
Quadratic QF1	28/22/-0.05000/0.0081	29/26/-0.05000/0.0055
Quadratic QF2	9/4/-1.01220/0.0045	9/4/-1.01220/0.0036
Ex. tridiagonal-2	13/11/3.50756/0.0057	24/22/3.50756/0.0040
Ex. three exponential terms	9/6/12.79633/0.0040	11/8/12.79633/0.0035

Table 2 Numerical results of PR+, NASQTR and NASCTR

Problem	n	PR+ Func/Iter/Fval	NASQTR Func/Iter/Fval	NASCTR Func/Iter/Fval
ARGLINA	200	5/5/2.00e+02	3/3/2.00e+02	3/3/2.00e+02
ARWHEAD	5000	15/15/0.00e+00	26/11/0.00e+00	25/10/5.55e-13
BDQRTIC	5000	123/123/2.00e+04	170/110/2.00e+04	148/102/2.00e+04
BOX	10000	19/19/-1.86e+03	57/20/-1.86e+03	54/20/-1.86e+03
BROWNAL	200	35/35/1.47e-09	25/9/1.47e-09	25/9/1.47e-09
BRYBND	5000	62/62/6.73e-12	44/32/1.48e-10	42/28/2.95e-11
CHNROSNB	50	562/562/3.79e-12	1234/1197/2.794e-11	1083/1055/5.09e-13
COSINE	10000	16/16/-1.00e+04	6/6/-1.00e+04	6/6/-1.00e+04
CRAGGLVY	5000	121/121/1.69e+03	447/282/1.69e+03	210/178/1.69e+03
DIXMAANA	3000	16/16/1.00e+00	10/7/1.00e+00	10/7/1.00e+00
DIXMAANB	3000	21/21/1.00e+00	11/7/1.00e+00	11/7/1.00e+00
DIXMAANC	3000	22/22/1.00e+00	13/8/1.00e+00	13/8/1.00e+00
DIXMAAND	3000	25/25/1.00e+00	15/9/1.00e+00	15/9/1.00e+00
DIXMAANE	3000	342/342/1.00e+00	209/206/1.00e+00	231/228/1.00e+00
DIXMAANF	3000	270/270/1.00e+00	320/316/1.00e+00	272/268/1.00e+00
DIXMAANG	3000	253/253/1.00e+00	264/259/1.00e+00	252/247/1.00e+00
DIXMAANH	3000	323/323/1.00e+00	250/244/1.00e+00	210/204/1.00e+00
DIXMAANI	3000	1074/1074/1.00e+00	580/573/1.00e+00	413/400/1.00e+00
DIXMAANJ	3000	242/242/1.00e+00	125/121/1.00e+00	155/151/1.00e+00
DIXMAANK	3000	219/219/1.00e+00	136/131/1.00e+00	174/169/1.00e+00
DIXMAANL	3000	203/203/1.00e+00	105/99/1.00e+00	108/102/1.00e+00
DIXON3DQ	100	203/203/2.27e-23	2818/2258/4.42e-06	1308/1118/2.91e-06
DQDRTIC	5000	15/15/1.24e-15	48/40/2.14e-15	48/40/2.14e-15
EDENSCH	2000	42/42/1.20e+04	23/15/1.20e+04	22/14/1.20e+04
ENGVAL1	5000	24/24/5.55e+03	18/12/5.55e+03	18/12/5.55e+03
EXTROSNB	10000	6643/6643/3.00e-06	105/89/4.15e-13	232/215/5.55e-03
FMINSRF2	5625	543/543/1.00e+00	1005/901/1.00e+00	850/793/1.00e+00
FMINSURF	1024	395/395/1.00e+00	628/592/1.00e+00	583/565/1.00e+00
GENHUMPS	100	3047/3047/4.72e-10	15971/9226/1.7307e-10	12625/8489/3.08e-11
GENROSE	100	595/595/1.00E+00	1053/880/1.00e+00	1207/1039/1.00e+00
MOREBV	5000	73/73/8.18e-10	36/36/2.55e-09	36/26/2.55e-09
NONDIA	5000	26/26/1.16e-09	41/20/9.21e-09	42/22/5.81e-10
NONDQUAR	5000	6686/6686/1.76e-06	2593/2316/6.21e-05	1924/1815/6.82e-05
POWELLSG	5000	342/342/9.10e-07	106/97/4.38e-05	97/88/2.66e-05
POWER	10000	669/669/3.08e-08	14261/3413/1.44e-08	1087/815/7.53e-09
SCHMVETT	5000	26/26/-1.50e+04	15/13/-1.50e+04	14/12/-1.49e+04
SENSORS	100	42/42/-2.11e+03	32/25/-2.11e+03	34/24/-2.11e+03
SINQUAD	5000	30/30/-6.76e+06	27/17/-6.76e+06	31/22/-6.76e+06
SPARSINE	100	483/483/4.80e-12	628/601/3.39e-12	595/585/3.02e-12
SPARSQUR	10000	122/122/5.93e-09	47/27/1.69e-07	35/18/4.43e-07
SPMSRTLS	4999	370/370/1.70e-09	370/369/2.65e-09	287/286/1.67e-09
SROSENBR	5000	27/27/2.16e-11	33/20/8.70e-12	32/20/2.19e-12

Table 2 (continued)

Problem	n	PR+ Func/Iter/Fval	NASQTR Func/Iter/Fval	NASCTR Func/Iter/Fval
TOINTGSS	5000	7/7/1.00e+01	3/2/1.00e+01	3/2/1.00e+01
WOODS	4000	343/343/6.69e-09	370/356/3.17e-09	409/395/4.26e-09

final objective function value (Fval), and the CPU time required in seconds (Time). The sign " – " means that when the number of iterations reaches 10000, the algorithm fails to stop.

From Table 1, we can see that the performance of NSCTR is competitive with NBCTR, and NSCTR spends less CPU time for most test problems. What's more, when the scale of a problem is more larger, NBCTR is quite costly and slowly, even fails. So, it is not suitable for large scale problems.

To analyze the feasibility and effectiveness of our simple conic trust region method for large scale problems, in the second experiment, our experiments are performed on a set of nonlinear unconstrained optimization problems from the CUTEr collection [2] with different dimensions ranging from 50 to 10000. The codes are written in Fortran 95 and in double precision arithmetic. All tests are performed on ASUS laptop (Intel Core2 Duo, 2.93GHz, 2GRAM) under Fedora 8 Linux and the gfortran compiler(version 4.1.2) with default options. The following three algorithms are implemented:

PR+: The conjugate gradient method in [8]. The Fortran codes (version 1.1) are obtained from J. Nocedal's web page at http://www.ece.northwestern.edu/\simnocedal/software.html.

NAQTR: The nonmonotone adaptive trust region method based on simple quadratic models proposed in [18].

NACTR: Our nonmonotone adaptive trust region method based on simple conic models, i.e., Algorithm 1.

Table 2 presents the numerical results. We consider the number of function evaluations and the number of iterations to compare the algorithms. Efficiency comparisons are made using the performance profile introduced by Dolan and Moré[3]. It is obvious that NASCTR seems to be more effective than PR+ and NASQTR in general (see Figs. 1 and 2).

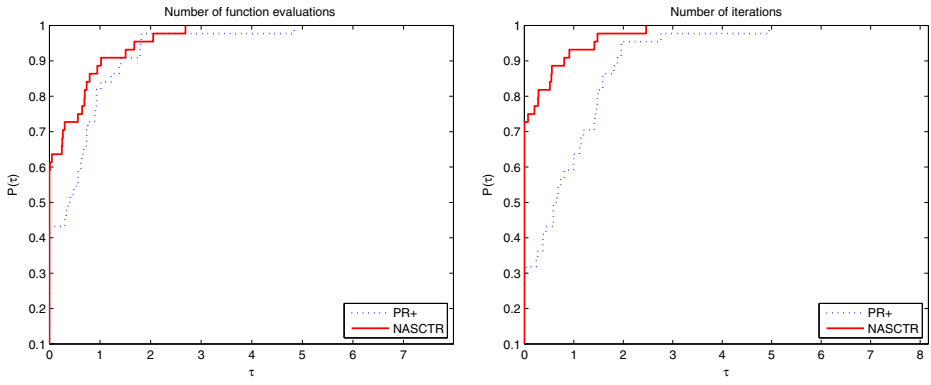

Fig. 1 Comparison of PR+ and NASCTR

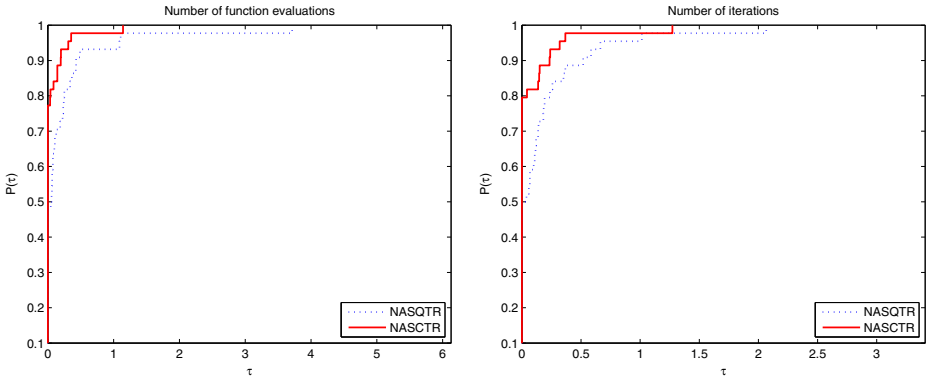

Fig. 2 Comparison of NASQTR and NASCTR

6 Conclusions

In this paper, we propose an adaptive trust region method based on simple conic models for unconstrained optimization and investigate its convergence. It only requires storage of first-order information during the process. Initial numerical results show that our new method is an improvement of the existing method and it is very suitable for large scale problems.

Acknowledgments The authors wish to thank two anonymous referees for their valuable comments and suggestions. This work is supported by National Natural Science Foundation of China(11471145), Natural Science Fund For Colleges and Universities in Jiangsu Province(13KJB110007) and the Foundation of Jiangsu University of Technology(KYY13012).

References

1. Andrei, N.: An unconstrained optimization test functions collection. Adv. Model. Optim. **10**, 147–161 (2008)
2. Bongartz, I., Conn, A.R., Gould, N.I.M., Toint, Ph.L.: CUTE: Constrained and Unconstrained Testing Environment. ACM T. Math. Software **21**, 123–160 (1995)
3. Dolan, E.D., Moré, J.J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)
4. Deng, N.Y., Xiao, Y., Zhou, F.J.: Nonmontonic trust region algorithm. J. Optim. Theory Appl. **26**, 259–285 (1993)
5. Fu, J., Sun, W., Sampaio, D.: An adaptive approach of conic trust-region method for unconstrained optimization problems. J. Appl. Math. Comput. **19**, 165–177 (2005)
6. Grippo, L., Lamparillo, F., Lucidi, S.: A nonmonotone line search technique for Newton's method. SIAM J. Numer. Anal. **23**, 707–716 (1986)
7. Gu, N., Mo, J.: Incorporating nonmonotone strategies into the trust region method for unconstrained optimization. Comput. Math. Appl. **55**, 2158–2172 (2008)
8. Gilbert, J.C., Nocedal, J.: Global convergence properties of conjugate gradient methods for optimization. SIAM J. Optim. **2**, 21–42 (1992)
9. Liu, P., Jiao, B., Chen, L.: A nonmonotonic trust region method based on the conic model. Adv. Math. **38**, 503–511 (2009)
10. Mo, J., Liu, C., Yan, S.: A nonmonotone trust region method based on nonincreasing technique of weighted average of the successive function values. J. Comput. Appl. Math. **209**, 97–108 (2007)
11. Qu, S., Jiang, S., Zhu, Y.: A conic trust-region method and its convergence properties. Comput. Math. Appl. **57**, 513–528 (2009)

12. Sang, Z., Sun, Q.: A self-adaptive trust region method with line search based on a simple subproblem model. J. Comput. Appl. Math. **232**, 514–522 (2009)
13. Wu, Q.: Nonmonotone trust region algorithm for unconstrained optimization problems. Appl. Math. Comput. **217**, 4274–4281 (2010)
14. Zhang, H.C., Hager, W.W.: A nonmonotone line search technique and its application to unconstrained optimization. SIAM J. Optim. **14**, 1043–1056 (2004)
15. Zhang, X., Zhang, J., Liao, L.: An adaptive trust region method and its convergence. Sci. China(Ser. A) **45**, 620–631 (2002)
16. Zhou, Q., Sun, W.: An adaptive nonmonotonic trust region method with curvilinear searches. J. Comput. Math. **24**, 761–770 (2006)
17. Zhu, M., Xue, Y., Zhang, F.: A quasi-Newton type trust region method based on the conic model. Numer. Math.: J. Chin. Univ. **17**, 36–47 (1995)
18. Zhou, Q., Zhang, C.: A new nonmonotone adaptive trust region method based on simple quadratic models. J. Appl. Math. Comput. **40**, 111–123 (2012)