Mathematical Modelling and Analysis Publisher: Taylor&Francis and VGTU
Volume 19 Number 4, September 2014, 469–490 http://www.tandfonline.com/TMMA
http://dx.doi.org/10.3846/13926292.2014.956237 Print ISSN: 1392-6292
Online ISSN: 1648-3510

An Improved Adaptive Trust-Region Method for Unconstrained Optimization

Hamid Esmaeili[a] and Morteza Kimiaei[b]

[a] *Department of Mathematics, Faculty of Science, Bu-Ali Sina University*
Hamedan, Iran
[b] *Department of Mathematics, Asadabad Branch, Islamic Azad University*
Asadabad, Iran
E-mail(*corresp.*): `esmaeili@basu.ac.ir`
E-mail: `Morteza.kimiaei@gmail.com`

Received June 1, 2013; revised July 23, 2014; published online September 1, 2014

Abstract. In this study, we propose a trust-region-based procedure to solve unconstrained optimization problems that take advantage of the nonmonotone technique to introduce an efficient adaptive radius strategy. In our approach, the adaptive technique leads to decreasing the total number of iterations, while utilizing the structure of nonmonotone formula helps us to handle large-scale problems. The new algorithm preserves the global convergence and has quadratic convergence under suitable conditions. Preliminary numerical experiments on standard test problems indicate the efficiency and robustness of the proposed approach for solving unconstrained optimization problems.

Keywords: unconstrained optimization, trust-region framework, nonmonotone technique, adaptive radius, convergence theory.

AMS Subject Classification: 90-08; 90C30; 90C90.

1 Introduction

Consider the following unconstrained optimization problem

$$\text{minimize } f(x), \quad \text{subject to } x \in \mathbf{R}^n, \tag{1.1}$$

where $f : \mathbf{R}^n \to \mathbf{R}$ is a continuous function. Many problems arising in science, engineering, management, economy and operations research can be reformulated into (1.1). A lot of approaches such as Newton, quasi-Newton, variable metric, gradient and conjugate gradient methods have been introduced to solve (1.1). These methods need to exploit one of the general globalization techniques, say line search or trust-region techniques, in order to guarantee the global convergence results (see [18]).

For a given iterate x_k, a line search technique refers to a procedure that computes a step length α_k along with a specific direction d_k and generates the new iterate

$$x_{k+1} = x_k + \alpha_k d_k.$$

Many line search strategies have been proposed to determine α_k; for instance, exact line search, Armijo rule, Wolfe and Goldstein inexact line search conditions (see [18]). On the other hand, a quadratic-based framework of trust-region technique computes a trial step d_k by solving the quadratic subproblem

$$\begin{aligned} &\text{minimize } m_k(x_k + d) = f_k + g_k^T d + \frac{1}{2} d^T B_k d \\ &\text{subject to } d \in \mathbf{R}^n \text{ and } \|d\| \le \Delta_k, \end{aligned} \tag{1.2}$$

where $\|\cdot\|$ denotes the Euclidean norm, $f_k = f(x_k)$, $g_k = \nabla f(x_k)$, B_k is the exact Hessian $G_k = \nabla^2 f(x_k)$ or its symmetric approximation and Δ_k is the trust-region radius. Let d_k be the solution of (1.2). In the traditional monotone trust-region methods, the ratio

$$r_k = \frac{f(x_k) - f(x_k + d_k)}{m_k(x_k) - m_k(x_k + d_k)} \tag{1.3}$$

between the actual reduction and the predicted reduction of $f(x)$ plays a key role in the algorithm to deciding whether the trial step d_k is acceptable and to adjust the new trust-region radius. Consider the step acceptance constants $0 < \mu_1 \le \mu_2 \le \mu_3 < 1$. It may be that $r_k \ge \mu_3$ (very successful iterate), $r_k \in [\mu_2, \mu_3)$ (not very successful iterate), $r_k \in [\mu_1, \mu_2)$ (successful iterate), or $r_k < \mu_1$ (unsuccessful iterate). In the first three cases, the trial step d_k is accepted, the next iterate x_{k+1} is chosen by $x_{k+1} = x_k + d_k$ and the trust-region radius is updated appropriately based on the amount of the ratio r_k. Finally, if iterate is unsuccessful, the trial step is rejected and the quadratic subproblem (1.2) would be solved again with the reduced trust-region radius.

It is known that the traditional trust-region methods are very sensitive to the initial radius Δ_0 and its updating scheme. This fact leads the researchers to work on finding appropriate procedures for initial radius as well as its updating rules. Sartenaer [20] gave a method which can automatically determine an initial trust-region radius. The drawback of this approach is the possible dependence of the parameters on the problem information. Recently, Gould et al. [12] extensively examined the sensitivity of the traditional trust-region methods to the parameters and trust-region radius updates. Despite their comprehensive tests on a large number of test problems, they could not claim to have found the best parameters to update scheme. In 2002, motivated by a problem in the neural network field, Zhang et al. [26] proposed the first adaptive trust-region radius in which the information of the current iterate was used more effectively to introduce the adaptive scheme. Precisely, they introduced the following adaptive trust-region radius

$$\Delta_k = c^{p_k} \|g_k\| \|\tilde{B}_k^{-1}\|,$$

where $c \in (0,1)$ is a constant, $p_k \in \mathbf{N}^* = \mathbf{N} \cup \{0\}$ and $\tilde{B}_k = B_k + E_k$ is a safely positive definite matrix based on Schnabel and Eskow scheme for modified Cholesky factorization, see [21]. According to numerical results, their method works very well on small-scale unconstrained optimization problems, but the situation dramatically changes for the large-scale and even medium-scale problems due to calculation of $\tilde{B}_k^{-1}$. Subsequently, Shi and Guo [23] proposed another interesting adaptive radius by

$$\Delta_k = -c^{p_k} \frac{g_k^T q_k}{q_k^T \hat{B}_k q_k} \|q_k\|. \tag{1.4}$$

Here, $\hat{B}_k = B_k + iI$, i is the smallest nonnegative integer such that $q_k^T \hat{B}_k q_k > 0$, I is identity matrix and q_k satisfies the well-known angle condition

$$-g_k^T q_k / (\|g_k\| \, \|q_k\|) \geq \tau,$$

in which $\tau \in (0,1]$ is a constant. An important advantage of (1.4) is its ability for selecting appropriate q_k in order to make a more robust method. They proposed the choices $-g_k$ and $-B_k^{-1} g_k$ for q_k. Preliminary numerical results along with theoretical analysis showed that their method was well promising to solve medium-scale unconstrained optimization problems without any need to search for appropriate initial trust-region radius.

Although the proposed adaptive trust-region radius by Shi and Guo enjoys some advantages such as decreasing the total computational cost by declining the number of subproblems to be solved and determining a good initial radius, it suffers from some drawbacks as well. We list some of these as follows:

- When $-g_k^T q_k$ is close to zero, we may obtain a tiny trust-region radius which results in increasing the total number of iterations.
- Due to the necessity of storing the matrix B_k for computing $q_k^T \hat{B}_k q_k$, this technique may be unsuitable for large-scale problems.
- The selection $q_k = -g_k$ does not generate an adequate radius (see [23]).
- Computation of $q_k = -B_k^{-1} g_k$ requires B_k^{-1} or solving a linear system of equations, so their method is not appropriate for large-scale problems.

The primary goal of the present paper is to propose an effective trust-region procedure for handling large-scale unconstrained optimization problems. So, we introduce a modified adaptive radius strategy based on a nonmonotone technique and employ it in a trust-region framework. The new method generates a suitable trust-region radius to decrease the total number of iterations for some of the test problems. We also investigate the global convergence to first-order stationary points and establish the quadratic convergence properties of the proposed algorithm. To illustrate the efficiency and robustness of our method, we report some numerical experiments.

The rest of this paper is organized as follows. In Section 2, we describe the motivation behind the proposed algorithm and its structure. Section 3 is devoted to investigating global and quadratic convergence properties of the algorithm. Numerical results are provided in Section 4 to show the well promising behavior of the proposed approach encountering with unconstrained optimization problems. Finally, some conclusions are outlined in Section 5.

2 New Algorithm: Motivation and Structure

A trust-region-based algorithm for solving unconstrained optimization will be presented in this section. After proposing an adaptive trust-region radius based on the nonmonotone technique, we add this strategy into trust-region framework to construct a more effective procedure for solving unconstrained optimization problems in the sequel.

Many researchers have investigated the disadvantages of traditional trust-region methods, especially those encountering with the rejected trial step, see [1, 12, 20, 23, 26]. If the elements of the trial step are close to the rejected one, the possibility of accepting the new trial step will be reduced significantly. Inspired by this fact, many researchers have worked on determining an appropriate trust-region radius and its updating rules, see [1, 12, 20]. However, there is no general rule to update the trust-region radius when iterate is successful or very successful. In these cases, it is necessary that the trust-region radius be appropriately large. However, if the trust-region is very large, the number of subproblems to be solved will be increased. Consequently, the computational cost of solving a problem may be increased, too. On the other hand, it is believed that a very small radius causes algorithm to increase the total number of iterates and decrease the efficiency of the procedure. Based on these ideas, to control the size of trust-region radius, we introduce a new adaptive trust-region which inherits some advantages of the nonmonotone technique.

To guarantee the global convergence of the traditional optimization approaches, it is well-known that we generally need to use a globalization technique, like line search or trust-region. These globalization techniques mostly enforce a monotonicity of the sequence of objective function values which usually result in producing short steps. Due to this fact, a slow numerical convergence is created with highly nonlinear problems, see [2, 3, 4, 11, 13, 14, 25]. In order to avoid this drawback of the Armijo-type line search globalization techniques, Grippo et al. [11] introduced a nonmonotone line search technique for Newton method. They relaxed Armijo rule such that stepsize α_k satisfied the following condition:

$$f(x_k + \alpha_k d_k) \leq f_{l(k)} + \delta \alpha_k g_k^T d_k, \tag{2.1}$$

where $\delta \in (0, 1)$ and

$$f_{l(k)} = \max_{0 \leq j \leq m(k)} \{f_{k-j}\}, \quad k \in \mathbf{N}^*, \tag{2.2}$$

in which $m(0) = 0$ and $0 \leq m(k) \leq \min\{m(k-1)+1, N\}$, with $N \geq 0$. The theoretical and numerical results show that (2.1) has remarkable positive effects on Armijo-type line searches to get faster global convergence especially for highly nonlinear problems. These excellent results are attracting many researchers to investigate more about the effects of these strategies on a wide variety of optimization procedures and propose new nonmonotone techniques, see [2, 3, 4, 11, 25]. As a prominent example, the first exploitation of nonmonotone strategies in a trust-region framework was proposed in [8] by changing the ratio (1.3) to assess an agreement between the quadratic model and the objective function over the trust-region area. More recently, the idea has been used by Ahookhosh and Amini [2] and Ahookhosh et al. [3] to introduce an algorithm for unconstrained optimization. These techniques employ the following nonmonotone term

$$R_k = \eta_k f_{l(k)} + (1-\eta_k) f_k, \tag{2.3}$$

where $\eta_k \in [\eta_{\min}, \eta_{\max}]$, $\eta_{\min} \in [0,1)$ and $\eta_{\max} \in [\eta_{\min}, 1]$. Clearly, the nonmonotonicity of (2.3) can be adjusted by selecting an adaptive process for η_k which makes it more relaxed for practical usage. As it was argued in [3, 7, 10] for an Armijo-type line search, it is generally believed that the best results can be obtained when a stronger nonmonotone term is used far away from the optimum while a weaker one is used close to the optimum. This also means that if the current iterate is far away from the optimum, a larger steplength will be used while being close to it, a smaller one can be employed. We believe the same idea is true for trust-region radius. Hence, we take the advantage of the nonmonotone technique to introduce a new adaptive radius by

$$\Delta_{k+1} = \begin{cases} \gamma_1 \Delta_k, & \text{if } r_k < \mu_1; \\ \max\{\gamma_2 \hat{R}_k, \Delta_k\}, & \text{if } r_k \in [\mu_1, \mu_2); \\ \hat{R}_k, & \text{if } r_k \in [\mu_2, \mu_3); \\ \max\{\gamma_3 \hat{R}_k, \Delta_k\}, & \text{if } r_k \geq \mu_3, \end{cases} \tag{2.4}$$

where γ_1, γ_2 and γ_3 are constants of trust-region scaling parameter,

$$\hat{R}_k = \eta_k g_{l(k)} + (1-\eta_k)\|g_k\| \tag{2.5}$$

and

$$g_{l(k)} = \max_{0 \leq j \leq m(k)} \big\{\|g_{k-j}\|\big\}, \quad k \in \mathbf{N}^*.$$

We note that if $\|g_{k+1}\| > g_{l(k)}$, the sequence $\{g_{l(k)}\}$ can not be descending, so that the descend condition $\|g_{k+1}\| \leq \|g_k\|$ may not be satisfied. As a result, this event causes the iterate to remain far away from optimum, due to the use of inappropriate information stored in $g_{l(k)}$ for generating the trust-region radius. To overcome this disadvantage, we design a removing procedure to eliminate the inappropriate members of the sequence $\{g_{l(k)}\}$. Our removing procedure works as follows: If $\|g_{k+1}\|$ is greater than $g_{l(k)}$, remove all elements of $g_{l(k)}$ and set $g_{l(k+1)} = \{\|g_{k+1}\|\}$. The modified sequence $\{g_{l(k)}\}$ has many benefits. We list some of them in the following:

- Removing unsuitable information of the sequence $\{g_{l(k)}\}$, we can generate a descending subsequence to produce an appropriate radius that leads to smaller steplength near optimum and greater steplength far away from it.

- The modified sequence $\{g_{l(k)}\}$ has descending subsequences that slowly shrink the trust-region radius and prevent the production of very small trust-region radius.

Now, we define the subsequence $\{g_{l(k)}\}_{k\in I}$ of $\{g_{l(k)}\}$ constructed of N-tuples $g_{l(k)}$ such that $i \in I$ results in

$$\|g_{i+1}\| \leq g_{l(i)}, \quad \text{for all } i \in I,$$

and terminate it when $\|g_{i+1}\| > g_{l(i)}$. Hence, the modified sequence $g_{l(k)}$ consists of a data structure whose mission is to store pertinent information and remove unsuitable information generated in the algorithm for determining the new adaptive radius. The proposed adaptive radius has many benefits. First of all, since $\hat{R}_k \geq \|g_k\|$ and the sequence $\{\hat{R}_k\}$ is not always decreasing, our new updating rule prevents the production of the very small trust-region radius as possible, so it decreases the total number of iterations for some of the test problems. Secondly, due to decreasing the subsequences of $\hat{R}_k$, $\varDelta_k$ will not stay too large, so the total number of subproblems to be solved will not be increased, either.

Now, we can outline our new adaptive trust-region-based algorithm as follows:

Algorithm 1: Adaptive Trust-Region Algorithm (ATRN)

Input: An initial point $x_0 \in \mathbf{R}^n$, a symmetric positive definite matrix $B_0 \in \mathbb{R}^{n\times n}$, k_{max}, $0 < \eta_0 < 1$,
$0 < \mu_1 \leq \mu_2 \leq \mu_3 < 1$, $0 < \gamma_1 \leq \gamma_2 < 1$, $\gamma_3 \geq 1$, $N > 0$ and $\epsilon > 0$.
Begin
  $\hat{R}_0 \leftarrow \|g_0\|$; $g_{l(0)} \leftarrow \|g_0\|$; $k \leftarrow 0$;
  While ($\|g_k\| \geq \epsilon$ **and** $k \leq k_{max}$) **{Start of outer loop}**
    $r_k \leftarrow 0$;
    While ($r_k < \mu_1$) **{Start of inner loop}**
      Step 1: {Step calculation}
      Specify the trial point d_k by solving the subproblem (1.2);
      Step 2: {Trial point acceptance}
      Determine the trust-region ratio r_k using (1.3);
      If $r_k < \mu_1$
        Update the trust-region radius by $\varDelta_k = \gamma_1 \varDelta_k$;
      End If
    End While {End of inner loop}
    $x_{k+1} \leftarrow x_k + d_k$;
    $f_{k+1} \leftarrow f(x_{k+1})$;
    $g_{k+1} \leftarrow g(x_{k+1})$;
    Step 3: {Trust-region radius update}
    If $\|g_{k+1}\| > g_{l(k)}$
      $m(k+1) \leftarrow 1$;
      $g_{l(k+1)} \leftarrow \|g_{k+1}\|$
    Else

$m(k+1) \leftarrow \min\{m(k)+1, N\}$;
Calculate $g_{l(k+1)}$ using (2.2);
End
Calculate $\hat{R}_{k+1}$ using (2.5);
Generate η_{k+1} by an adaptive formula;
Update Δ_{k+1} using (2.4);
Step 4: {Parameters update}
Update B_{k+1} by a quasi-Newton formula;
$k \leftarrow k+1$;
End While {End of outer loop}
End

---

3 Theoretical Results Analysis

This section is devoted to analyzing the convergence properties of Algorithm 1. We first give some properties of the algorithm and then investigate its global convergence to first-order critical points. The quadratic convergence rates of the proposed algorithm are also considered in this section.

Throughout the paper, we consider the following assumptions in order to analyze the convergence of the new algorithm:

(H1) The objective function $f(x)$ is continuously differentiable and the level set $L(x_0) = \{x \in \mathbf{R}^n \mid f(x) \leq f(x_0)\}$ is bounded for any $x_0 \in \mathbf{R}^n$.

(H2) The objective function $g(x)$ is continuously differentiable and the set $\mathcal{F}(x_0) = \{x \in \mathbf{R}^n \mid \|g(x)\| \leq \|g(x_0)\|\}$ is bounded for any $x_0 \in \mathbf{R}^n$.

(H3) The approximation Hessian matrix B_k is uniformly bounded, i.e., there exists a constant $M > 1$ such that $\|B_k\| \leq M$, for all $k \in \mathbf{N}^*$.

Remark 1. If the objective function $f(x)$ is twice continuously differentiable and the level set $L(x_0)$ is bounded, (H1) implies that $\|\nabla^2 f(x)\|$ is uniformly continuous and bounded above on an open bounded convex set Ω, containing $L(x_0)$. As a result, there exists a constant $L > 0$ such that $\|\nabla^2 f(x)\| \leq L$, for all $x \in \Omega$. Therefore, using mean value theorem, one can conclude that

$$\big\|g(x) - g(y)\big\| \leq L\|x - y\|, \quad \forall x, y \in \Omega,$$

which means that $g(x)$ is Lipschitz continuous in the Ω.

Remark 2. To establish the global convergence property, we assume that the decrease on the model m_k is at least as much as a fraction of the one obtained by Cauchy point, i.e. there exists a constant $\beta \in (0,1)$ such that

$$m_k(x_k) - m_k(x_k + d_k) \geq \beta\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\} \quad \forall k. \tag{3.1}$$

The relation (3.1) is called the sufficient reduction condition. It implies that $d_k \neq 0$ whenever $g_k \neq 0$.

Lemma 1. *Suppose that (H3) holds and the sequence* $\{x_k\}$ *is generated by Algorithm* 1. *Then, we have*

$$\big|f(x_k) - f(x_k + d_k) - \big(m_k(x_k) - m_k(x_k + d_k)\big)\big| \le O\big(\|d_k\|^2\big).$$

Proof. Taylor expansion along with (H3) implies that

$$\begin{aligned}&\big|f_k - f(x_k + d_k) - \big(m_k(x_k) - m_k(x_k + d_k)\big)\big|\\ &\quad \le \big|-d_k^T G_k d_k + d_k^T B_k d_k\big| + O\big(\|d_k\|^2\big) = \big|d_k^T (B_k - G_k) d_k\big| + O\big(\|d_k\|^2\big)\\ &\quad \le (L + M)\|d_k\|^2 + O\big(\|d_k\|^2\big) = O\big(\|d_k\|^2\big).\end{aligned}$$

This completes the proof. □

In the following lemma, we show that the subsequence $\{g_{l(k)}\}_{k\in I}$ is decreasing. This leads to producing a small or large trust-region radius being close to or far away from the optimum, respectively.

Lemma 2. *Suppose that the sequence* $\{x_k\}$ *is generated by Algorithm* 1. *Then, for all* $k \in \mathbf{N}^* \cap I$, *we have* $x_k \in \mathcal{F}(x_0)$ *and* $\{g_{l(k)}\}_{k\in I}$ *is a decreasing subsequence of* $\{g_{l(k)}\}$.

Proof. Using the definition of $\hat{R}_k$ and $g_{l(k)}$, we have

$$\|g_k\| \le \hat{R}_k \le g_{l(k)}. \tag{3.2}$$

To prove that $\{g_{l(k)}\}_{k\in I}$ is decreasing, we consider two following cases.

i) $k \ge N$. In this case, we have $m(k+1) \le m(k) + 1$, for all $k \in I$. So, the definition of $g_{l(k+1)}$ and removing procedure result in

$$\begin{aligned} g_{l(k+1)} &= \max_{0\le j\le m(k+1)} \big\{\|g_{k+1-j}\|\big\}\\ &\le \max\Big\{\max_{0\le j\le m(k)} \big\{\|g_{k-j}\|\big\}, \|g_{k+1}\|\Big\}\\ &= \max\big\{g_{l(k)}, \|g_{k+1}\|\big\} = g_{l(k)}.\end{aligned}$$

ii) $k < N$. In this case, $m(k) = k$, for all $k \in I$, and using $\|g_k\| \le \|g_0\|$, we can see that

$$g_{l(k)} = \|g_0\| \quad \forall k \in \mathbf{N}^* \cap I.$$

Now we prove that $x_k \in \mathcal{F}(x_0)$, for all $k \in \mathbf{N}^* \cap I$. Obviously, the definition of $\hat{R}_k$ indicates that $\hat{R}_0 = \|g_0\|$. By induction, assume that $x_i \in \mathcal{F}(x_0)$, for all $i = 1, \ldots, k$. Using (3.2) and part one of the proof, we obtain

$$\|g_{k+1}\| \le \hat{R}_{k+1} \le g_{l(k+1)} \le g_{l(k)} \le \|g_0\|,$$

that completes the proof. □

The following lemma will establish that the sequence $\{g_{l(k)}\}$ is convergent.

Lemma 3. *Suppose that (H1)–(H3) and Remark 1 hold and there exists a constant $\kappa > 0$ such that $\kappa\|g_k\| > \|d_k\|$. Assume, furthermore, that the sequence $\{x_k\}$ is generated by Algorithm 1. Then, we have*

$$\lim_{k\to\infty} g_{l(k)} = \lim_{k\to\infty} \big\|g(x_k)\big\|.$$

Proof. It is followed from the definition of x_{k+1} and $f_{l(k)}$ that

$$f_{l(k)} - f(x_k + d_k) \geq f_k - f(x_k + d_k) \geq \mu_1\big[m_k(x_k) - m_k(x_k + d_k)\big].$$

By substituting the index k with $l(k) - 1$, we get

$$f_{l(l(k)-1)} - f_{l(k)} \geq \mu_1\big[m_k(x_{l(k)-1}) - m_k(x_{l(k)})\big],$$

so

$$\lim_{k\to\infty}\big[m_k(x_{l(k)-1}) - m_k(x_{l(k)})\big] = 0. \tag{3.3}$$

On the other hand, according to (H3) and (3.1), we have

$$\begin{aligned} m_k(x_{l(k)-1}) - m_k(x_{l(k)}) &\geq \beta\|g_{l(k)-1}\| \min\left\{\Delta_{l(k)-1}, \frac{\|g_{l(k)-1}\|}{B_{l(k)-1}}\right\} \\ &\geq \beta\|g_{l(k)-1}\| \min\left\{\|d_{l(k)-1}\|, \frac{\|d_{l(k)-1}\|}{\kappa M}\right\} \\ &\geq \frac{\beta}{\kappa}\min\left\{1, \frac{1}{\kappa M}\right\}\|d_{l(k)-1}\|^2 = \zeta\|d_{l(k)-1}\|^2 \geq 0, \end{aligned}$$

where $\zeta = \frac{\beta}{\kappa}\min\left\{1, \frac{1}{\kappa M}\right\}$. The above inequality and (3.3) imply that

$$\lim_{k\to\infty}\|d_{l(k)-1}\| = 0. \tag{3.4}$$

Lipschitz continuity of $g(x)$ along with (3.4) results in

$$\lim_{k\to\infty}\big\|g(x_{l(k)})\big\| = \lim_{k\to\infty}\big\|g(x_{l(k)-1})\big\|. \tag{3.5}$$

Similar to [11], we define $\hat{l}(k) = l(k + N + 2)$. By induction, for all $j \geq 1$, we show

$$\lim_{k\to\infty}\|d_{\hat{l}(k)-j}\| = 0. \tag{3.6}$$

For $j = 1$, (3.6) follows from (3.4) because $\{\hat{l}(k)\} \subset \{l(k)\}$. Assuming that (3.6) holds for a given j, we show that it holds for $j + 1$, too. Let k be sufficiently large such that $\hat{l}(k) - (j + 1) > 0$. Using Lemma 7 of [2] and substituting k with $\hat{l}(k) - j - 1$, we have

$$f(x_{\hat{l}(k)-j-1}) - f(x_{\hat{l}(k)-j}) \geq \mu_1\big[m_k(x_{\hat{l}(k)-j-1}) - m_k(x_{\hat{l}(k)-j})\big].$$

Following the same argument to derive (3.4), we deduce

$$\lim_{k\to\infty}\big\|d_{\hat{l}(k)-j-1}\big\| = 0.$$

This means that the induction is completed and (3.6) holds for any $j \geq 1$. Similar to (3.5), for any given $j \geq 1$, we have that

$$\lim_{k\to\infty} \|g(x_{\hat{l}(k)-j})\| = \lim_{k\to\infty} \|g(x_{l(k)})\|.$$

On the other hand, for any k, one can write

$$x_{k+1} = x_{\hat{l}(k)} - \sum_{j=1}^{\hat{l}(k)-k-1} d_{\hat{l}(k)-j},$$

which along with (3.6) and the fact $\hat{l}(k) - j - 1 \leq N + 1$ implies that

$$\lim_{k\to\infty} \|x_{k+1} - x_{\hat{l}(k)}\| = 0.$$

Therefore, from the Lipschitz continuity of $g(x)$, we get

$$\lim_{k\to\infty} g_{l(k)} = \lim_{k\to\infty} \|g(x_{l(k)})\| = \lim_{k\to\infty} \|g(x_{\hat{l}(k)})\| = \lim_{k\to\infty} \|g(x_k)\|,$$

that completes the proof. □

Corollary 1. Suppose that sequence $\{x_k\}$ is generated by Algorithm 1. Then,

$$\lim_{k\to\infty} \hat{R}_k = \lim_{k\to\infty} \|g(x_k)\|.$$

Lemma 4. *Suppose that (H2)–(H3) hold, the sequence* $\{x_k\}$ *is generated by Algorithm* 1, *and* d_k *is a solution of the subproblem* (1.2). *Then, there exists a constant* L' *such that*

$$m_k(x_k) - m_k(x_k + d_k) \geq L'\|g_k\|^2. \tag{3.7}$$

Proof. Using (H3), (3.1) and (2.4), we have

$$\begin{aligned} m_k(x_k) - m_k(x_k + d_k) &\geq \beta\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\} \\ &\geq \beta\|g_k\| \min\left\{\gamma_3 \hat{R}_{k-1}, \frac{\|g_k\|}{M}\right\} \geq \beta\|g_k\| \min\left\{\gamma_3 \hat{R}_k, \frac{\|g_k\|}{M}\right\} \\ &\geq \beta\|g_k\| \min\left\{\hat{R}_k, \frac{\|g_k\|}{M}\right\} \geq \beta\|g_k\| \min\left\{\|g_k\|, \frac{\|g_k\|}{M}\right\} = L'\|g_k\|^2, \end{aligned}$$

where $L' = \beta \min\left\{1, \frac{1}{M}\right\}$. □

The earliest proofs of first-order convergence are those of Powell for unconstrained optimization [19], the person who proved that $\lim_{k\to\infty} \inf \|g_k\| = 0$. This result was extended by Thomas [24] through proving that under additional conditions, $\lim_{k\to\infty} \|g_k\| = 0$. Though, Thomas's proof intensely relies on Powell's result. The Powell's theorem has been called a remarkable one, not just because of the fact that it demands weak assumptions on f, but also for his proof presents a general framework in order to prove the convergence of trust region algorithms. An algorithm must contain two following properties to be fit within the framework:

(P1) If $\|g_k\|$ is bounded away from zero and $\{x_k\}$ converges, then $\Delta_k \nrightarrow 0$.

(P2) If $f(x_k)$ is bounded below and $\|g_k\|$ is bounded away from zero, then $\Delta_k \to 0$ and $\{x_k\}$ converges.

It follows immediately that for any algorithm satisfying (P1) and (P2), either $f(x_k)$ is unbounded below or $\lim_{k\to\infty} \inf \|g_k\| = 0$.

The two following lemmas show that our algorithm satisfy to P1 and P2.

Lemma 5. *If $\|g_k\| \geq \epsilon$ for all k, then Δ_k does not converge to 0.*

Proof. By contradiction, for all sufficiently large k, assume that $\Delta_k \to 0$. Suppose that there is a subsequence $\{\|g_k\|\}_{k\in\mathcal{J}}$ such that $\|g_k\| \leq \Delta_k M$ for all $k \in \mathcal{J}$. Then, we obtain

$$\Delta_k \geq \frac{\|g_k\|}{M} \geq \frac{\epsilon}{M}.$$

Consequently, Δ_k does not converge to zero. Hence, Suppose that there is a positive index k_0 such that

$$\|g_k\| \geq \Delta_k M \tag{3.8}$$

for all $k \geq k_0$. Using Lemma 4 and (3.8), for all $k \geq k_0$, we have

$$m_k(x_k) - m_k(x_k + d_k) \geq L'\|g_k\|^2 \geq L'M^2\Delta_k^2,$$

so

$$\begin{aligned} |r_k - 1| &= \left| \frac{f(x_k + d_k) - m_k(x_k + d_k)}{m_k(x_k) - m_k(x_k + d_k)} \right| \\ &\leq \frac{O(\|d_k\|^2)}{L'M^2\Delta_k^2} \leq \frac{O(\Delta_k^2)}{L'M^2\Delta_k^2}. \end{aligned}$$

Therefore, for all sufficiently large k, $|r_k - 1| < 1 - \mu_3$, or $r_k \geq \mu_3$. Hence, we can conclude that

$$\Delta_{k+1} = \max\{\gamma_3 \hat{R}_k, \Delta_k\} \geq \gamma_3 \hat{R}_k \geq \gamma_3\|g_k\| \geq \bar{\epsilon},$$

where $\gamma_3\epsilon = \bar{\epsilon}$. This contradicts our working assumption that $\Delta_k \to 0$. □

Lemma 6. *If $\{f(x_k)\}$ is bounded below and $\|g_k\| > \epsilon$ for all k, then $\Delta_k \to 0$ and the sequence $\{x_k\}$ converges.*

Proof. Take $\mathbf{K_1} = \{k \mid r_k \geq \mu_2\}$ and $\mathbf{K_2} = \{k \in \mathbf{K_1} \mid \Delta_k \geq \frac{\|g_k\|}{M}\}$. By Lemma 4, for $k \in \mathbf{K_2}$,

$$f_k - f_{k+1} \geq \mu_2(m_k(x_k) - m_k(x_k + d_k)) \geq L'\mu_2\|g_k\|^2 > \sigma_1\epsilon^2,$$

where $\sigma_1 = L'\mu_2$. Since $\{f_k\}$ is convergent, we have

$$\sum_{k\in\mathbf{K_2}} \sigma_1\epsilon^2 \leq \sum_{k\in\mathbf{K_2}} (f_k - f_{k+1}) \leq \sum_k (f_k - f_{k+1}) < \infty.$$

Therefore, $\mathbf{K_2}$ must be finite. As a result, there exists a $k_0 \in \mathbf{N}$ such that $\Delta_k \leq \frac{\|g_k\|}{M}$ for all $k \geq k_0$ and $k \in \mathbf{K_1}$. By setting $\mathbf{K_3} = \{k \geq k_0 \mid \Delta_k \leq \frac{\|g_k\|}{M}, k \in \mathbf{K_1}\}$, we have

$$f_k - f_{k+1} \geq \mu_2\big(m_k(x_k) - m_k(x_k + d_k)\big) \geq L'\mu_2\|g_k\|^2 \geq \sigma_1 M^2 \Delta_k^2 = \sigma_2 \Delta_k^2$$

for $k \in \mathbf{K_3}$. Since $\{f_k\}$ is convergent, with $\sigma_2 = \sigma_1 M^2$, we obtain

$$\sum_{k \in \mathbf{K_3}} \Delta_k \leq \frac{1}{\sigma_2} \sum_{k \in \mathbf{K_3}} (f_k - f_{k+1}) \leq \frac{1}{\sigma_2} \sum_k (f_k - f_{k+1}) < \infty.$$

Hence, $\mathbf{K_3}$ must be finite. For $k \notin \mathbf{K_1}$, according to Algorithm 1 and (H3), we have

$$\begin{aligned}\Delta_{k+1} &\leq \max\{\gamma_2 \hat{R}_k, \Delta_k\} \leq \max\left\{\gamma_2 \hat{R}_k, \frac{\|g_k\|}{M}\right\} \\ &\leq \max\left\{\gamma_2 \hat{R}_k, \frac{\hat{R}_k}{M}\right\} = \hat{R}_k \max\left\{\gamma_2, \frac{1}{M}\right\} = \pi \hat{R}_k,\end{aligned}$$

where $\pi = \max\{\gamma_2, \frac{1}{M}\}$ belongs to $(0,1)$. Based on this information, we rewrite the set $\mathbf{K_3}$ as follows $\mathbf{K_3} = \{k_1, k_2, \ldots, k_j, \ldots\}$, where $k_1 < k_2 < \cdots < k_j < \cdots$. Therefore, if $k_j \in \mathbf{K_3} - \mathbf{K_1}$, we have

$$\sum_{k_j < k < k_{j+1}} \Delta_k \leq \sum_{i=0}^{k_{j+1}-k_j} \pi^i \hat{R}_{k_j} \leq \frac{1}{1-\pi} \hat{R}_{k_j}.$$

Therefore,

$$\begin{aligned}\sum_{k \geq k_0} \|x_{k+1} - x_k\|_\infty &\leq \sum_{k \geq k_0} \Delta_k = \sum_{k \in \mathbf{K_3}} \Delta_k + \sum_{k \notin \mathbf{K_3}} \Delta_k \\ &= \sum_{k \in \mathbf{K_3}} \Delta_k + \sum_{j=1}^{\infty} \sum_{k_j < k < k_{j+1}} \Delta_k \leq \sum_{k_j \in \mathbf{K_3}} \Delta_{k_j} + \frac{1}{1-\pi} \sum_{j=1}^{\infty} \hat{R}_{k_j} \\ &= \sum_{k_j \in \mathbf{K_3}} \Delta_{k_j} + \frac{1}{1-\pi} \sum_{k_j \in \mathbf{K_3}} \hat{R}_{k_j} < \infty,\end{aligned}$$

which implies that $\{x_k\}$ is a Cauchy sequence and $\Delta_k \to 0$. $\square$

Assumptions (H1)–(H3) and the combination of the two previous lemmas imply the following result.

Corollary 2. Suppose that (H1)–(H3) hold. Then, Algorithm 1 either stops at a stationary point of $f(x)$ or generates an infinite sequence $\{x_k\}$ such that

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

In the following theorem, we prove that Algorithm 1 is globally convergent to the first-order critical points under the mentioned assumptions.

Theorem 1. *Suppose that (H1)–(H3) hold. Then, Algorithm* 1 *either stops at a stationary point of* $f(x)$ *or generates an infinite sequence* $\{x_k\}$ *such that*

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

Proof. Assuming that $\{x_k\}$ is not finitely terminating, we show that the equality $\lim_{k\to\infty} \|g_k\|=0$ is valid. By contradiction, for all sufficiently large k, suppose that there exists a constant $\epsilon > 0$ and an infinite subset $K \subseteq \mathbf{N}^*$ such that

$$\|g_k\| > \epsilon \quad \text{for all } k \in K. \tag{3.9}$$

Using (3.7), (3.9) and $r_k > \mu_1$, we can write

$$f_k - f(x_k + d_k) \geq \mu_1 \big[m_k(x_k) - m_k(x_k + d_k)\big] \geq \mu_1 L' \ \|g_k\|^2 \geq \mu_1 \epsilon^2 L' > 0.$$

This fact along with Lemma 6 imply that $f_k - f(x_k + d_k) \to 0$ for all sufficiently large k. Now, by taking a limit we get $\mu_1 \epsilon^2 L' \leq 0$ which is a contradiction. Hence, this completes the proof. □

To establish that Algorithm 1 is quadratically convergent some additional assumptions are further required. These conditions can be stated as follows:

(H4) There exist some constants $c_0 > 0$ and $\rho_1 \in (0,1)$ such that

$$\big\|g(x) - g(y) + G(y)(x-y)\big\| \leq c_0 \|x-y\|^2 \quad \text{for all } x, y \in N(x_*, \rho_1),$$

where x_* is a solution of (1.1) and $N(x_*, \rho_1) = \{x \mid \|x - x_*\| \leq \rho_1\}$.

(H5) There exist some constants $c_1 \geq \frac{1}{\gamma_2}$ and $\rho_2 \in (0,1)$ such that

$$c_1 \|x - x_*\| \leq \big\|g(x)\big\| = \big\|g(x) - g(x_*)\big\| \quad \text{for all } x \in N(x_*, \rho_2),$$

where x_* is a solution of (1.1) and $N(x_*, \rho_2) = \{x \mid \|x - x_*\| \leq \rho_2\}$.

In the sequel, we simply choose $\rho = \min\{\rho_1, \rho_2\}$. The condition (H4) holds if $g(x)$ and $G(x)$ are continuously differentiable and Lipschitz continuous, respectively .

Theorem 2. *Suppose that (H1)–(H5) hold and let the sequence* $\{x_k\}$*, generated by Algorithm* 1*, converge to* x_**. Then, for sufficiently large* k*, we have*

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

where d_k *is a solution of* (1.2)*. Furthermore, the sequence* $\{x_k\}$ *converges quadratically to* x_**.*

Proof. If d_k is a solution of (1.2), then we first show that $x_{k+1} = x_k + d_k$, for sufficiently large k. From the fact that d_k is a feasible point for the subproblem (1.2), Corollary 1 and Theorem 1, we simply have

$$\|d_k\| \leq \Delta_k \leq \gamma_3 \hat{R}_{k_j} \to 0, \quad \text{as} \quad k \to \infty, \tag{3.10}$$

where $0 \leq k_j \leq k$. Note that $\|g_k\| \geq \epsilon$ because Algorithm 1 is not stopped. This fact together with Lemma 1, Lemma 4 and (3.10) suggests that

$$\begin{aligned}\left|\frac{f_k - f(x_k+d_k)}{m_k(x_k) - m_k(x_k+d_k)} - 1\right| &= \left|\frac{m_k(x_k+d_k) - f(x_k+d_k)}{m_k(x_k) - m_k(x_k+d_k)}\right| \\ &\leq \frac{O(\|d_k\|^2)}{L'\|F_k\|^2} \leq \frac{O((\Delta_k)^2)}{L'\epsilon^2} \to 0, \quad \text{as} \quad k \to \infty.\end{aligned}$$

So, for sufficiently large k, we have $r_k \geq \mu_1$ meaning that the trial point d_k is accepted by Algorithm 1.

At this point, the quadratic convergence of the sequence $\{x_k\}$ generated by Algorithm 1 is investigated. Regarding (H1), it is obvious that the level set $L(x_0)$ is bounded and $g(x)$ is continuously differentiable on the compact convex set Ω containing $L(x_0)$. Therefore, there exists a constant $M_0 > 0$ such that

$$\|G_k\| \leq M_0 \quad \text{for all } x \in \Omega. \tag{3.11}$$

Hence, from (3.11) and the mean value theorem, one can easily get

$$\|g_k\| = \big\|g_k - g(x_*)\big\| \leq \big\|G(\xi)\big\|\|x_k - x_*\| \leq M_0\|x_k - x_*\|$$

for all $x_k \in N(x_*, \rho)$ and $\xi \in [x_k, x_*]$. As a result, we can write

$$\hat{R}_k \approx \|g_k\| \leq M_0\|x_k - x_*\|$$

for all sufficiently large k and

$$\begin{aligned}\|d_k\| &\leq \Delta_k \leq \max\{\gamma_3\hat{R}_k, \Delta_k\} \\ &\leq \gamma_3\hat{R}_{k_j} \leq \gamma_3 M_0\|x_k - x_*\|,\end{aligned} \tag{3.12}$$

where $0 \leq k_j \leq k$. To show that the point $x_k - x_*$ is a feasible point for (1.2), we consider the three following cases.

(a) If $r_k \in [\mu_1, \mu_2)$, (H5) result in

$$\|x_k - x_*\| \leq \frac{1}{c_1}\|g_k\| \leq \frac{1}{c_1}\hat{R}_k \leq \gamma_2\hat{R}_k \leq \max\{\gamma_2\hat{R}_k, \Delta_k\} = \Delta_k.$$

(b) If $r_k \in [\mu_2, \mu_3)$, from (H5), we have

$$\|x_k - x_*\| \leq \frac{1}{c_1}\|g_k\| \leq \frac{1}{c_1}\hat{R}_k \leq \hat{R}_k = \Delta_k.$$

(c) If $r_k \geq \mu_3$, then (H5) implies that

$$\|x_k - x_*\| \leq \frac{1}{c_1}\|g_k\| \leq \frac{1}{c_1}\hat{R}_k \leq \hat{R}_k \leq \max\{\gamma_3\hat{R}_k, \Delta_k\} = \Delta_k.$$

Here, from (H4), (H5) and (3.12), we can conclude that

$$\begin{aligned}c_1\|x_{k+1} - x_*\| &\leq \big\|g(x_k+d_k)\big\| \leq \|g_k + G_k d_k\| + O\big(\|d_k\|^2\big) \\ &= \|g_k - g_* + G_k d_k\| + O\big(\|d_k\|^2\big) \\ &\leq O\big(\|x_k - x_*\|^2\big) + O\big(\|x_k - x_*\|^2\big) = O\big(\|x_k - x_*\|^2\big).\end{aligned}$$

Thus, there exists a positive constant κ such that

$$\lim_{k\to\infty} \frac{\|x_{k+1} - x_*\|}{\|x_k - x_*\|^2} = \lim_{k\to\infty} \frac{O(\|x_k - x_*\|^2)}{\|x_k - x_*\|^2} \le \kappa.$$

Therefore, the sequence $\{x_k\}$, generated by Algorithm 1, is quadratically convergent. □

4 Preliminary Numerical Experiments

We now report the numerical results obtained by running Algorithms (ATRN-1 and ATRN-2) in comparison with the traditional trust-region algorithm (TTR) and the adaptive trust-region algorithm of Shi and Guo [23] with $q_k = -H_k g_k$ (ATRS) on 93 standard unconstrained test problems. In Table 1, problems are taken from Lukšan and Vlček [16, 17]. For all of the above algorithms, the trust-region subproblems are coded due to Steihaug–Toint procedure, see [7]. The Steihaug-Toint algorithm terminates at $x_k + d$ when

$$\big\|\nabla m(x_k + d)\big\| \le \min\big\{0.01, \big\|\nabla m_k(x_k)\big\|^{\frac{1}{2}}\big\}\big\|\nabla m_k(x_k)\big\| \quad \text{or} \quad \|d\| = \Delta_k,$$

holds. All codes are written in MATLAB 9 programming environment with double precision format in the same subroutine. In our numerical experiments, the algorithms are stopped when $\|g_k\| \le 10^{-6}\sqrt{n}$ or the total number of iterates exceeds 20000. The latter case is denoted as “Failed” in the presented table. During the code implementation, we verified whether the different codes converged to the same point. We only provided data for problems in which all algorithms converged to the identical point. In all algorithms, the matrix B_k is updated by the following compact limited memory BFGS formula

$$B_k = B_k^{(0)} - \begin{bmatrix} Y_k & B_k^{(0)} S_k \end{bmatrix} \begin{bmatrix} -D_k & L_k^T \\ L_k & S_k^T B_k^{(0)} S_k \end{bmatrix}^{-1} \begin{bmatrix} Y_k^T \\ S_k^T B_k^{(0)} \end{bmatrix},$$

where $B_k^{(0)} = \lambda I$, for some positive scalar λ, and the matrices S_k, Y_k, D_k and L_k are defined as follows:

$$\begin{aligned} S_k &= [s_{k-m}, \ldots, s_{k-1}], \quad Y_k = [y_{k-m}, \ldots, y_{k-1}], \\ D_k &= \operatorname{diag}\big[s_{k-m}^T y_{k-m}, \ldots, s_{k-1}^T y_{k-1}\big], \\ (L_k)_{i,j} &= \begin{cases} s_{k-m+i-1}^T y_{k-m+j-1}, & \text{if } i > j \\ 0 & \text{otherwise,} \end{cases} \end{aligned}$$

in which $s_k = x_{k+1} - x_k$, $y_k = g_{k+1} - g_k$ and $m = \min\{k, m_1\}$. In our implementation, we take

$$\lambda = \frac{\|y^{k_m}\|^2}{{y^{k_m}}^T s^{k_m}},$$

as suggested by Shanno and Phua [22]. However, we do not update B_k whenever the curvature condition, i.e. $s_{k_i}^T y_{k_i} > 0$ for $i = 1, \ldots, m$, does not hold, see

[6, 15]. The code of the compact limited memory BFGS updating formula is rewritten based on ASTRAL code from J. V. Burke in [5]. The common parameters of the algorithms TTR, ATRN-1 and ATRN-2 are set to $\mu_1 = 10^{-5}$, $\mu_2 = 0.2$, $\mu_3 = 0.8$, $\gamma_1 = 0.25$, $\gamma_2 = 0.5$, $\gamma_3 = 2$, and $m_1 = 5$, similar to [5]. In the ATRN-1 and ATRN-2 algorithm, the parameter η_k is updated by

$$\eta_k = \begin{cases} \eta_0/2, & \text{if } k = 1; \\ (\eta_{k-1} + \eta_{k-2})/2, & \text{if } k \geq 2, \end{cases}$$

while the trust-region radius is updated by

$$\Delta_{k+1} = \begin{cases} \gamma_1 \|d_k\|, & \text{if } r_k < \mu_1; \\ \max\{\gamma_2 \hat{R}_k, \Delta_k\}, & \text{if } r_k \in [\mu_1, \mu_2); \\ \hat{R}_k, & \text{if } r_k \in [\mu_2, \mu_3); \\ \max\{\gamma_3 \hat{R}_k, \Delta_k\}, & \text{if } r_k \geq \mu_3, \end{cases}$$

where $\eta_0 = 0.95$ and $\eta_0 = 0.85$ are chosen for ATRN-1 and ATRN-2 algorithms, respectively. Furthermore, we use $N = 10$ in these algorithms.

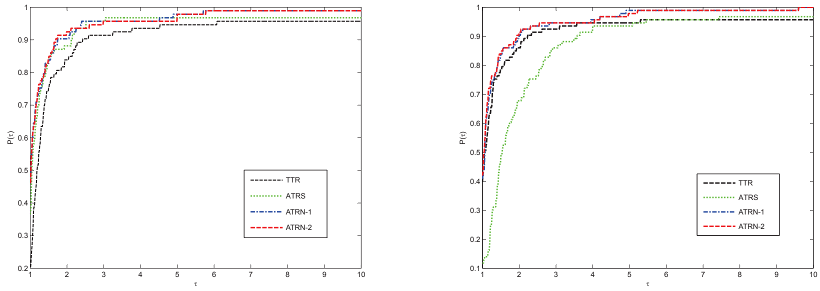
The TTR algorithm employs $\Delta_0 = 10$ and updates the trust-region radius by

$$\Delta_{k+1} = \begin{cases} \gamma_1 \|d_k\|, & \text{if } r_k < \mu_1; \\ \max\{\gamma_2 \|d_k\|, \Delta_k\}, & \text{if } r_k \in [\mu_1, \mu_2); \\ \Delta_k, & \text{if } r_k \in [\mu_2, \mu_3); \\ \max\{\gamma_3 \|d_k\|, \Delta_k\}, & \text{if } r_k \geq \mu_3. \end{cases}$$

Due to [23], ATRS algorithm employs $c = 0.75$, $\mu = 0.1$ and calculates the $q_k = -H_k g_k$ using the algorithm QN in [15].

Notice that in all algorithms, the total number of iterates, N_i, is identical to that of gradient evaluations, N_g. Due to this fact, in Table 1, we have just reported the number of iterates and the number of function evaluations, N_f, as a performance measure for the algorithms. It can be seen from Table 1 that in most cases ATRN-1 and ATRN-2 are remarkably better than other considered algorithms in both the number of iterates and function evaluations. Although the ATRN-1 and ATRN-2 are not the best in some problems, it usually has better computational performance compared with other algorithms. We also take advantages of the performance profile of Dolan and Moré [9] to have a better comparison among considered algorithms. Therefore, we have illustrated the results of Table 1 in Figures 1 according to the total number of iterates and the total number of function evaluations, respectively. In these figures, P designates to the percentage of problems which are solved within a factor τ of the best solver.

From Figure 1 (a) , firstly, it can be easily seen that ATRN-1 and ATRN-2 have most wins among all other considered algorithms. More precisely, it solves about 49% of the test problems more efficiently and is faster than others. Secondly, the performance of ATRN-1 and ATRN-2 are better than TTR and ATRS in the sense of the total number of iterates. Thirdly, considering the ability of completing the run successfully, we observe that both of ATRN-1 and



a) b)

Figure 1. Results for the presented algorithm: a) iteration performance profiles, b) function evaluations performance profiles .

ATRN-2 are the best among other considered algorithms. Finally, the performance index of ATRN-1 and ATRN-2 grows up faster than other. This means that whenever ATRN-1 and ATRN-2 are not the best, theirs performance index are close to that of the best algorithm. On the other hand, Figure 1 (b) shows that ATRN-1, ATRN-2 and TTR are so competitive regarding the total number of function evaluations; however, they perform better than ATRS. Moreover, the results of ATRN-1 and ATRN-2 have most wins in about 40% of test problems. In a final word, our preliminary computational experiments show that the ATRN-1 and ATRN-2 algorithms with both amounts of mentioned η_0 are remarkably well-promising for solving large-scale unconstrained optimization problems.

Table 1. Numerical results

Problem name	Dim	TTR		ATRS		ATRN-1		ATRN-2	
		N_i	N_f	N_i	N_f	N_i	N_f	N_i	N_f
1. ERRINROS MODIFIED	5	Failed	Failed	1978	5797	3084	4226	2070	2827
2. Brent	10	8127	9303	4411	8816	6215	7346	6215	7346
3. Flow in a channel	10	121	143	146	249	88	110	88	110
4. Tridiagonal exponential	10	216	248	126	223	196	229	196	229
5. FREUROTH	10	31	40	26	85	23	40	23	40
6. Troesch	50	5781	6063	5417	6842	4591	4775	4940	5219
7. INDEF	100	126	154	93	250	161	196	127	148
8. Tridiagonal system	100	1655	1965	Failed	Failed	740	923	740	923
9. NONCVXUN	300	778	813	582	686	692	724	692	724
10. Trigonometric-exponential system 2	300	4501	5024	9902	12712	5496	6154	5496	6154
11. Potra and Rheinboldt boundery value	300	761	762	1917	2413	3436	3608	3803	3976
12. Toint trigonometric	500	2723	2814	1223	1518	1198	1263	1198	1263
13. Problem 206	500	4266	4463	8403	10629	3924	4002	4023	4091
14. Chained serpentine	500	6807	8302	6451	15155	6581	9566	6563	9579
15. Five-diagonal system	500	5181	6109	129	199	120	143	120	143
16. Modified discrete boundary value	500	5511	5742	5249	6536	4698	4795	4497	4558
17. Chained wood	500	4757	5413	Failed	Failed	2330	2812	3942	4817
18. Problem 201	500	12532	12848	14552	17131	9765	10177	9989	10408
19. NONCVXU2	500	8206	8553	9652	11958	3419	3527	3173	3274
20. NCB20B CORRECTED	500	2955	3250	2329	3238	2280	2561	2280	2561
21. Countercurrent reactors 2	504	Failed	Failed	117	177	98	113	100	118
22. SINQUAD	1000	181	209	179	608	207	311	208	281
23. CHAINWOO	1000	5136	5842	283	527	4128	5056	4128	5056
24. CHROSNB MODIFIED	1000	6017	6809	5886	9777	5830	7150	5830	7150
25. Chained Rosenbrock	1000	6393	7733	5907	13426	5971	8339	5971	8339
26. Another trigonometric	1000	17	18	17	25	18	19	18	19
27. Problem 213	1000	4020	4165	4361	5272	4181	4254	3096	3179
28. Ascher and Russel boundery value	1000	4134	4340	5271	6638	2815	2885	2614	2703
29. Chained exponential	1000	106	110	111	150	108	120	108	120
30. Structured Jacobian	1000	265	320	560	1277	435	590	435	590
31. Seven-diagonal system	1000	242	255	170	233	113	125	113	125
32. Seven-diag. gen. of the broyden tridiag.	1000	43	39	55	136	35	39	35	39
33. Generalization of the Brown function 1	1000	368	398	261	427	261	307	261	307
34. Extended Freudenstein and Roth	1000	68	78	41	109	28	42	28	42
35. Banded trigonometric	1000	196	204	209	244	206	217	206	217
36. Variational 1	1000	1782	1859	1667	2052	1626	1701	1626	1701
37. Variational 2	1000	2484	2626	2203	2694	2123	2247	2123	2247

Table 1. Numerical results (*continued*)

38. Variational 3	1000	2232	2308	2341	2793	2521	2661	2521	2661
39. Variational 4	1000	2596	2686	2661	3192	2419	2535	2419	2535
40. Variational Calvar 2	1000	5363	5519	5597	6788	4356	4519	4356	4519
41. GENHUMPS	1200	2490	2802	2027	5446	1948	2415	2010	2474
42. GENROSE	1200	2912	2438	2853	6603	2809	3802	2809	3802
43. FLETCHCR	1200	7094	7559	5803	7305	6290	7332	6290	7332
44. CRAGGLVY	3000	96	105	83	146	84	100	84	100
45. DIXMAANE	3000	218	230	204	224	225	437	250	260
46. DIXMAANF	3000	213	225	138	175	143	148	160	171
47. DIXMAANG	3000	204	213	146	184	142	146	204	212
48. DIXMAANH	3000	182	189	132	147	173	181	210	222
49. DIXMAANI	3000	1402	1466	723	844	4125	4143	4174	4187
50. DIXMAANJ	3000	169	177	198	214	190	201	159	170
51. DIXMAANK	3000	186	196	148	185	198	208	170	179
52. DIXMAANL	3000	153	162	153	201	153	160	135	143
53. DIXMAANM	3000	1351	1409	1133	1405	5533	5547	5660	5677
54. DIXMAANN	3000	364	381	390	492	872	880	768	781
55. DIXMAANO	3000	347	363	332	382	418	427	396	404
56. DIXMAANP	3000	304	310	304	353	324	334	298	307
57. TOINTGSS	5000	23	29	14	16	16	20	16	20
58. TQUARTIC	5000	64	73	33	364	74	138	42	49
59. WOODS	5000	267	331	150	383	71	107	185	247
60. NONDQUAR	5000	545	623	501	932	1189	1265	1495	1584
61. POWELLSG	5000	109	139	52	208	50	68	50	68
62. Chained powell singular	5000	86	99	84	211	118	141	118	141
63. Trigonometric-exponential system 1	5000	27	35	23	50	23	36	23	36
64. Generalization of the Brown function 2	5000	13	14	9	17	4	6	4	6
65. Discrete boundary value	5000	1	2	1	2	1	2	1	2
66. Problem 202	5000	8	9	7	10	7	8	7	8
67. Problem 207	5000	218	231	289	354	233	246	241	257
68. Problem 208	5000	29	33	30	54	30	40	30	40
69. Extended Rosenbrock	5000	61	73	70	207	63	112	56	89
70. Extended Powell Singular	5000	62	77	43	115	64	87	64	87
71. Broyden tridiagonal (problem 36)	5000	37	41	39	66	29	35	29	35
72. Generalized Broyden tridiagonal	5000	16	18	381	464	20	26	20	26
73. Generalized Broyden Banded	5000	43	46	43	67	45	50	45	50
74. Toint quadratic merging	5000	57	61	63	162	51	69	51	69
75. Attracting-Repelling	5000	144	158	134	215	143	170	143	170
76. Singular Broyden	5000	98	111	117	180	201	232	195	215
77. Extended Gragg and Levy	5000	53	64	41	62	34	41	34	41

Table 1. Numerical results (*continued*)

78. Broyden tridiagonal (problem 62)	5000	51	56	74	78	77	78	77	78
79. Extended Wood	5000	28	35	27	68	24	38	24	38
80. ARWHEAD	10000	9	12	9	48	8	17	8	17
81. BROYDN7D	10000	77	82	23	58	17	23	17	23
82. COSINE	10000	85	90	14	25	14	17	14	17
83. DQRTIC	10000	97	99	85	159	82	98	82	98
84. EDENSCH	10000	19	22	20	35	14	18	14	18
85. EG2	10000	4	7	4	35	7	21	7	21
86. SROSENBR	10000	74	96	186	488	122	192	122	192
87. ENGVAL1	20000	21	24	19	41	15	23	15	23
88. EXTROSNB	20000	21	27	22	52	21	29	21	29
89. LIARWHD	20000	48	52	32	96	32	45	32	45
90. MOREBV DIFFERENT START POINT	20000	27	32	25	34	23	28	23	28
91. NONDIA	20000	9	11	8	60	9	21	9	21
92. SCHMVETT	20000	20	25	22	29	19	28	19	28
93. SPARSQUR	20000	46	47	49	104	47	61	55	67

5 Concluding Remarks

In this paper, we presented a trust-region method for solving unconstrained optimization problems in which an adaptive radius is proposed based on nonmonotone technique. The new adaptive procedure increases the trust-region radius to find the optimum in a larger region. Consequently, it decreases the total number of iterations and therefore it will decrease the total number of subproblems to be solved. From the theoretical analysis point of view, the proposed algorithm inherits the global convergence of traditional trust-region algorithms to first-order critical points under classical assumptions. Under some suitable conditions, the quadratic convergence rate is established. Finally, our preliminary numerical experiments on a large set of standard test problems point out that the proposed algorithm is remarkably efficient and robust for solving large-scale unconstrained optimization problems.

References

[1] M. Ahookhosh and K. Amini. A nonmonotone trust region method with adaptive radius for unconstrained optimization problems. *Comput. Math. Appl.*, **60**:411–422, 2010. http://dx.doi.org/10.1016/j.camwa.2010.04.034.

[2] M. Ahookhosh and K. Amini. An efficient nonmonotone trust-region method for unconstrained optimization. *Numer. Algorithms*, **59**:523–540, 2012. http://dx.doi.org/10.1007/s11075-011-9502-5.

[3] M. Ahookhosh, K. Amini and M.R. Peyghami. A nonmonotone trust-region line search method for large-scale unconstrained optimization. *Appl. Math. Model.*, **36**:478–487, 2012. http://dx.doi.org/10.1016/j.apm.2011.07.021.

[4] M. Ahookhosh, H. Esmaeili and M. Kimiaei. An effective trust-region-based approach for symmetric nonlinear systems. *Int. J. Comput. Math.*, **90**:671–690, 2013. http://dx.doi.org/10.1080/00207160.2012.736617.

[5] J.V. Burke and L. Xu. An active set ℓ_∞-trust region algorithm for box constrained optimization. Technical report preprint, University of Washington, 2007. Available from Internet: `http://www.optimization-online.org/DB_HTML/2007/07/1717.html`.

[6] R. Byrd, J. Nocedal and R. Schnabel. Representation of quasi-newton matrices and their use in limited memory methods. *Math. Program.*, **63**:129–156, 1994. http://dx.doi.org/10.1007/BF01582063.

[7] A.R. Conn, N.I.M. Gould and Ph.L. Toint. *Trust-Region Methods.* Society for Industrial and Applied Mathematics, SIAM, Philadelphia, 2000. ISBN 0521517753.

[8] N.Y. Deng, Y. Xiao and F.J. Zhou. Nonmonotonic trust region algorithm. *J. Optim. Theory Appl.*, **26**:259–285, 1993. http://dx.doi.org/10.1007/BF00939608.

[9] E.D. Dolan and J.J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, **91**:201–213, 2002. http://dx.doi.org/10.1007/s101070100263.

[10] H. Esmaeili and M. Kimiaei. A new adaptive trust-region method for system of nonlinear equations. *Appl. Math. Model.*, **38**(11–12):3003–3015, 2014. http://dx.doi.org/10.1016/j.apm.2013.11.023.

[11] G. Fasano, F. Lampariello and S. Lucidi. A nonmonotone line search technique for Newton's method. *SIAM J. Numer. Anal.*, **23**:707–716, 1986. http://dx.doi.org/10.1137/0723046.

[12] N. Gould, D. Orban, A. Sartenaer and Ph L. Toint. Sentesivity of trust region algorithms to their parameters. *4OR. A Quarterly Journal of Operations Research*, **3**:227–241, 2005.

[13] L. Grippo, F. Lampariello and S. Lucidi. A truncated newton method with nonmonotone linesearch for unconstrained optimization. *J. Optim. Theory Appl.*, **60**(3):401–419, 1989. http://dx.doi.org/10.1007/BF00940345.

[14] L. Grippo, F. Lampariello and S. Lucidi. A class of nonmonotone stabilization method in unconstrained optimization. *Numer. Math.*, **59**:779–805, 1991. http://dx.doi.org/10.1007/BF01385810.

[15] L. Kaufman. Reduced storage, quasi-newton trust region approaches to function optimization. *SIAM J. Optim.*, **10**(1):56–69, 1999. http://dx.doi.org/10.1137/S1052623496303779.

[16] L. Lukšan, C. Matonoha and J. Vlček. Modified cute problems for sparse unconstrained optimization. Techical Report V-1081, ICS AS CR, Prague, 2010.

[17] L. Lukšan and C. Vlček. Sparse test problems for unconstrained optimization. Techical Report V-1064, ICS AS CR, Prague, 2003.

[18] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer, New York, 2006. ISBN 0387303030.

[19] M.J.D. Powell. On the global convergence of trust region algorithms for unconstrained optimization. *Math. Program.*, **29**:297–303, 1984. http://dx.doi.org/10.1007/BF02591998.

[20] A. Sartenaer. Automatic determination of an initial trust region in nonlinear programming. *SIAM J. Sci. Comput.*, **18**(6):1788–1803, 1997. http://dx.doi.org/10.1137/S1064827595286955.

[21] R.B. Schnabel and E. Eskow. A new modified Cholesky factorization. *SIAM J. Sci. Comput.*, **11**(6):1136–1158, 1990. http://dx.doi.org/10.1137/0911064.

[22] D.F. Shanno and K.H. Phua. Matrix conditioning and non-linear optimization. *Math. Program.*, **14**:149–160, 1978. http://dx.doi.org/10.1007/BF01588962.

[23] Z.J. Shi and J.H. Guo. A new trust region method with adaptive radius. *Comput. Optim. Appl.*, **41**:225–242, 2008. http://dx.doi.org/10.1007/s10589-007-9099-8.

[24] S.W. Thomas. *Sequential Estimation Techniques for Quasi-Newton Algorithms.* Cornell University, 1975.

[25] H.C. Zhang and W.W. Hager. A nonmonotone line search technique for unconstrained optimization. *SIAM J. Optim.*, **14**(4):1043–1056, 2004. http://dx.doi.org/10.1137/S1052623403428208.

[26] X.S. Zhang, J.L. Zhang and L.Z. Liao. An adaptive trust region method and its convergence. *Sci. China*, **45**:620–631, 2002.