Journal of Mathematical Modeling
Vol. 13, No. 1, 2025, pp. 219–233. Research Article

JMM

Incorporating non-monotone trust region algorithm with line search method for unconstrained optimization

Seyed Hamzeh Mirzaei, Ali Ashrafi*

Department of Mathematics, Statistics and Computer Science, Semnan University, Semnan, Iran
Email(s): shmirzaei@semnan.ac.ir, a_ashrafi@semnan.ac.ir

Abstract. This paper concerns an efficient trust region framework that exploits a new non-monotone line search method. The new algorithm avoids the sudden increase of the objective function values in the non-monotone trust region method. Instead of resolving the trust region subproblem whenever the trial step is rejected, the proposed algorithm employs an Armijo-type line search method in the direction of the rejected trial step to construct a new point. Global and superlinear properties are preserved under appropriate conditions. Comparative numerical experiments depict the efficiency and robustness of the new algorithm using the Dolan-More performance profiles.

Keywords: Unconstrained optimization, trust region, line search, non-monotone technique.
AMS Subject Classification 2010: 90C30, 90C53, 49M37.

1 Introduction

Here, the following unconstrained optimization problem is considered:

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

where $f : \mathbb{R}^n \to \mathbb{R}$ is a twice continuously differentiable function. Among the various methods that have been presented to solve problem (1), two of the most famous categories are line search (LS) and trust region (TR) methods [6, 18]. The sequence of iterations in LS and TR methods is as follows:

$$x_{k+1} = x_k + p_k, \quad k = 1, 2, 3, ...,$$

where $x_0 \in \mathbb{R}^n$ is a given initial point. In the LS method $p_k = \alpha_k d_k$, where, d_k is the search direction and α_k is called the step length. Several inexact LS methods have been proposed, see for example [4, 12, 24].

*Corresponding author
Received: 23 August 2024 / Revised: 8 November 2024 / Accepted: 12 November 2024
DOI: 10.22124/jmm.2024.28257.2492

On the other hand, in the TR method, p_k is called the trial step, which is obtained by solving the following subproblem:

$$\begin{aligned} &\min_{p\in\mathbb{R}^n} \psi_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p, \\ &s.t. \qquad \|p\| \leq \Delta_k, \end{aligned} \tag{2}$$

where $\psi_k(p)$ is an approximate model of f, $f_k = f(x_k)$, $g_k = \nabla f(x_k)$, $B_k \in \mathbb{R}^{n\times n}$ is an approximation of exact Hessian $H_k = \nabla^2 f(x_k)$ and $\Delta_k > 0$ is the TR radius that is updated in each iteration. We refer to $\|\cdot\|$ as the Euclidean norm.

The agreement between the objective function f and model $\psi_k(p)$ is measured by the so-called TR ratio r_k defined as follows:

$$r_k = \frac{f_k - f(x_k + p_k)}{\psi_k(0) - \psi_k(p_k)}.$$

The r_k value, helps us to decide whether to accept or reject the trial step p_k. If r_k is close to unity, then the trial step p_k is accepted, $x_{k+1} = x_k + p_k$ and the algorithm can expand the TR radius for the next iteration. Conversely, if r_k is a positive number close to zero or even negative, we conclude that there is no reasonable agreement between the exact and the approximate model. Therefore, the trial step is rejected and the TR radius should be shrunk. Consequently, the TR subproblem must be solved again. Hence, the computational cost of the algorithm may increase. To overcome this defect, a new class of TR algorithms incorporating LS rules has been developed that avoids resolving the TR subproblem [11, 19]. Although this combination had favorable results, it has been shown in [13, 14, 22] that when the sequence $\{x_k\}$ is trapped near the presence of narrow curved valleys, the monotone LS becomes obsolete. To overcome this shortcoming, the use of non-monotone techniques seems very meaningful.

Perhaps the first non-monotone technique is the "watchdog technique" proposed by Chamberlain et al. [5]. Based on this scheme, Grippo et al. [13] proposed a non-monotone LS method in the following form:

$$f(x_k + \alpha_k p_k) \leq f_{l(k)} + \sigma \alpha_k g_k^T p_k,$$

in which $\sigma \in (0,1)$,

$$f_{l(k)} = \max_{0\leq j\leq \phi(k)} \{f_{k-j}\}, \quad k = 0,1,2,...,$$

$\phi(0) = 0$, $0 \leq \phi(k) \leq \min\{\phi(k-1)+1, N\}$ for all $k \geq 1$ and $N \geq 0$. Despite the good advantages of this technique, Zhang and Hager [26] found that this method suffers from various weaknesses. Therefore a non-monotone technique based on the weighted average of previous consecutive iterations was proposed by them. Their method can overcome the shortcomings of previous methods. However, the numerical results presented in [15, 26] indicate that updating some parameters becomes an encumbrance at each iteration.

The first exploitation of the non-monotone technique in the TR framework was presented by Deng et al. [7]. Further development of this technique was done by [22, 25, 27]. The basic difference between the monotone and non-monotone TR algorithms is due to the definition of the ratio r_k. The most popular non-monotone TR ratio is defined as follows:

$$\hat{r}_k = \frac{f_{l(k)} - f(x_k + p_k)}{\psi_k(0) - \psi_k(p_k)}. \tag{3}$$

Since the factor $f_{l(k)}$ only appears in the numerator, the non-monotone ratio (3) may increase considerably, especially for large values of N. So, by placing $f_{l(k)}$ in the denominator, Fu and Sun [10] provided more reasonable information about the exact and approximate models. Some modified versions of the non-monotone TR ratio have been proposed, see for example [1, 8].

We know that by taking the maximum in the parameter $f_{l(k)}$, it is possible to ignore a good function value. In trying to deal with this defect, a non-monotone parameter was introduced by Ahookhosh and Amini [2] as follows:

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

where $\eta_k \in [\eta_{min}, \eta_{max}]$ for $\eta_{min} \in [0,1)$ and $\eta_{max} \in [\eta_{min}, 1]$. Taking advantage of the non-monotone term of in [2], a modified non-monotone TR method with Armijo-type LS method was presented by Rezaee and Babaie-Kafaki [20]. Kamandi and Amini [16] found that in some problems, for example, OSCIGRAD, the difference between the current value f_k and the non-monotone parameter R_k becomes too large and in this case, a large increase is allowed to happen in the next iteration. Another disadvantage of the above non-monotone technique is its strong dependence on the choice of memory parameter $\phi(k)$ and parameter η_k. However, there is no specific rule to adjust them.

Considering the mentioned weaknesses, in this study, we propose a new non-monotone TR combined with the LS method, which prevents the sudden increase of the objective function values in the non-monotone TR method, but does not depend on the choice of $\phi(k)$ and η_k. The properties of global convergence and superlinearity are preserved under appropriate conditions.

The remainder of this paper is arranged as follows. Description of the new algorithm is given in Section 2. Analysis of global and superlinear properties are discussed in Section 3. Preliminary numerical results are reported in Section 4, and finally conclusions are given in Section 5.

2 New algorithm

This section is dedicated to describing the structure of the new algorithm. Similar to [16], first we define sequence $\{Q_k\}$ as follows:

$$Q_k = \begin{cases} 0, & \text{if } \ k=0, \text{ or } f_{l(k)} - f_k > \nu |f_k|, \\ Q_{k-1}+1, & \text{o.w}, \end{cases}$$

where ν is a positive parameter. When the relative difference between $f_{l(k)}$ and the value of the current function is large, it makes the algorithm monotone, which prevents the sudden increase of the values of the objective function for the next iteration. Second, we define sequence $\{I_k\}$ as follows:

$$I_k = \begin{cases} 0, & \text{if } \ k=0, \text{ or } f_k < f_{k-1}, \\ I_{k-1}+1, & \text{o.w.} \end{cases}$$

The sequence $\{I_k\}$ counts the number of successive increments in the values of the objective function. Moreover, having the above sequences for fixed natural numbers $\bar{N}$ and $\bar{I}$, we define the new non-monotone parameter D_k as:

$$D_k = \begin{cases} \max\limits_{0\le j\le n_k} \{f_{k-j}\}, & \text{if } \ I_k \le \bar{I}, \\ f(x_k), & \text{o.w}, \end{cases} \tag{4}$$

where $n_k = \min\{Q_k, \bar{N}\}$. The new non-monotone parameter not only prevents the sudden increase of objective function values but also does not depend on the choice of $\phi(k)$ and η_k. Finally, we define the TR ratio as follows:

$$\hat{r}_k = \frac{D_k - f(x_k + p_k)}{\psi_k(0) - \psi_k(p_k)}, \tag{5}$$

so that, D_k is updated by (4).

To avoid resolving the TR subproblem, here we use a non-monotone version of the Armijo-type LS procedure proposed by Wan et al. [23]. Based on this LS, we calculate α_k as the largest quantity in the set $\{s_k, \rho s_k, \rho^2 s_k, ...\}$, which satisfies the following inequality:

$$f(x_k + \alpha_k p_k) \leq D_k + \sigma \alpha_k \left(g_k^T p_k - \frac{1}{2} \alpha_k \ell L_k \|p_k\|^2 \right), \tag{6}$$

where D_k is updated by (4), $s_k = -g_k^T p_k / L_k \|p_k\|^2$ is a positive constant, $\rho \in (0,1), \sigma \in (0, \frac{1}{2}), \ell \in [\,0, +\infty)$ and the parameter L_k is an approximation of the Lipschitz constant, which can be estimated by $L_k = \|y_{k-1}\| / \|p_{k-1}\|$, in which $y_{k-1} = g_k - g_{k-1}$. If $\ell = 0$ and $D_k = f_k$, then the new LS rule (6) reduces to the Armijo LS rule [4].

The general framework of the new algorithm is as follows:

Algorithm 1 Non-monotone trust region line search algorithm (NTRLS)

Inputs: Give initial point $x_0 \in \mathbb{R}^n$, positive definite matrix $B_0 \in \mathbb{R}^{n \times n}$, $\varepsilon > 0$, $0 < \mu_0 < 1$, $0 < c_1 < 1 < c_2$, $\bar{N}, \bar{I} > 0$, $\Delta_0 > 0$, $\rho \in (0,1)$, $\sigma \in (0, \frac{1}{2})$, $\ell \in [\,0, +\infty)$ and $L_0 > 0$. Compute $f(x_0)$ and set $k = 0$.
Step 1: Compute g_k. If $\|g_k\| < \varepsilon$, then stop.
Step 2: Solve subproblem (2) to find the trial step p_k.
Step 3: Compute $\hat{r}_k$ by (5). If $\hat{r}_k \geq \mu_0$, go to Step 5. Otherwise go to Step 4.
Step 4: Find step length α_k satisfying (6) and set $x_{k+1} = x_k + \alpha_k p_k$. Update the TR radius by $\Delta_{k+1} \in [c_1 \|x_{k+1} - x_k\|, \Delta_k]$, and go to Step 6.
Step 5: Set $x_{k+1} = x_k + p_k$ and $\Delta_{k+1} = c_2 \Delta_k$.
Step 6: Calculate the new B_{k+1} using the quasi-Newton updating formula.
Step 7: Set $k = k + 1$ and go to Step 1.

It is worth noting that the quasi-Newton updating formula is explained in the numerical experiments in Section 4.

3 Convergence analysis

Here, we turn to discuss the convergence properties of Algorithm 1. To this end, we require the following assumptions [2].

Assumption 1. *The level set* $\Gamma(x_0) = \{x | f(x) \leq f(x_0)\} \subset \Im$, *where* $\Im$ *is a closed and bounded subset of* $\mathbb{R}^n$.

Assumption 2. *There exists a positive constant λ such that*

$$p^T B_k p \geq \lambda \|p\|^2, \quad \forall p \in \mathbb{R}^n \text{ and } k \in \mathbb{N}.$$

Assumption 3. *The matrix B_k is uniformly bounded, i.e., there exists a constant $\Lambda_1 > 0$ such that $\|B_k\| \leq \Lambda_1$, $\forall k \in \mathbb{N}$.*

Remark 1. *Let f be a twice continuously differentiable function. Therefore Assumption 1 shows that there exists a constant $\Lambda_2 > 0$, such that*

$$\|\nabla^2 f(x_k)\| \leq \Lambda_2, \quad \forall x \in \Im. \tag{7}$$

Furthermore, from the mean value theorem, it is concluded that

$$\|g(x) - g(y)\| \leq \Lambda_2 \|x - y\|, \quad \forall x, y \in \Im, \tag{8}$$

which implies that $g(x)$ is Lipschitz continues in the $\Im$. Consequently, from (8) for the sequence $\{L_k\}$, we have $0 \leq L_k \leq \Lambda_2$.

At each iteration, a trial step p_k is generated by solving the TR subproblem (2). Strong theoretical and implementation results for the proposed algorithm can be obtained if the trial step p_k satisfies

$$\psi_k(0) - \psi_k(p_k) \geq \gamma \|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}, \qquad \forall\, k \geq 0, \tag{9}$$

and

$$g_k^T p_k \leq -\gamma \|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}, \qquad \forall\, k \geq 0, \tag{10}$$

where $\gamma \in (0,1)$ is a constant. Similar to [19], we can solve the TR subproblem inaccurately so that (9) and (10) hold.

For the remainder of this paper, we suppose that Assumptions 1–3 are true. Moreover, we use two index set $I = \{k : \hat{r}_k \geq \mu_0\}$ and $J = \{k : \hat{r}_k < \mu_0\}$, in order to ease of operations.

Lemma 1. *Let $\{x_k\}$ be the sequence generated by Algorithm 1. Then for all k we have $f_{k+1} \leq D_k$.*

Proof. We consider two cases:

Case 1. $k \in I$. From (5) and (9), we can write $f_{k+1} \leq D_k - \mu_0 \gamma \|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}$, therefore, we obtain $f_{k+1} \leq D_k$, for all $k \in I$.

Case 2. $k \in J$. Through (10) we know that $g_k^T p_k \leq 0$, for all k. Therefore, from this inequality along with (6), we conclude

$$f_{k+1} - D_k \leq \sigma \alpha_k \left(g_k^T p_k - \frac{1}{2}\alpha_k \ell L_k \|p_k\|^2\right) \leq \sigma \alpha_k g_k^T p_k \leq 0,$$

hence, we have $f_{k+1} \leq D_k$, for all $k \in J$. □

Lemma 2. *The sequence $\{x_k\}$ generated by Algorithm 1 is contained in the level set $\Gamma(x_0)$ and the sequence $\{D_k\}$ is convergent.*

Proof. The proof is similar to the proof of [16, Lemma 3.6] and thus it is omitted. □

Lemma 3. *Step 4 of Algorithm 1 is well-defined.*

Proof. If $k \in I$, the proof is straightforward. Let $k \in J$, for the purpose of deriving a contradiction, suppose that there exists $k \in J$ such that

$$f(x_k + \rho^j s_k p_k) \geq D_k + \sigma \rho^j s_k \big(g_k^T p_k - \frac{1}{2}\rho^j s_k \ell L_k \|p_k\|^2\big), \quad \forall j \in \mathbb{N} \cup \{0\}.$$

From Lemmas 1 and 2, we obtain

$$\frac{f(x_k + \rho^j s_k p_k) - f_k}{\rho^j s_k} > \sigma \big(g_k^T p_k - \frac{1}{2}\rho^j s_k \ell L_k \|p_k\|^2\big).$$

Since f is differentiable and $\rho \in (0,1)$, by taking the limit with $j \to \infty$, we get

$$g_k^T p_k \geq \sigma g_k^T p_k. \tag{11}$$

Due to the fact that $\sigma \in (0, \frac{1}{2})$, inequality (11) leads us to $g_k^T p_k \geq 0$, which contradicts (10). This implies that for any $k \in J$, there exists a $j_k > 0$, such that (6) holds [15]. □

Lemma 4. *For all $k \in J$ the step length α_k satisfies*

$$\alpha_k > \frac{(1-\sigma)\rho\lambda}{\Lambda_2(1+\sigma\ell)}.$$

Proof. Assume $\alpha = \frac{\alpha_k}{\rho}$, then from Step 4 of Algorithm 1, we can write

$$f(x_k + \alpha p_k) > D_k + \sigma\alpha \big(g_k^T p_k - \frac{1}{2}\alpha \ell L_k \|p_k\|^2\big). \tag{12}$$

By Taylors expansion, we have

$$f(x_k + \alpha p_k) = f_k + \alpha g_k^T p_k + \frac{1}{2}\alpha^2 p_k^T \nabla^2 f(\zeta_k) p_k, \tag{13}$$

where $\zeta_k \in (x_k, x_k + \alpha p_k)$. From (7), (12) and (13) together with Lemmas 1 and 2, we obtain

$$\sigma\alpha \big(g_k^T p_k - \frac{1}{2}\alpha \ell L_k \|p_k\|^2\big) < \alpha g_k^T p_k + \frac{1}{2}\alpha^2 \Lambda_2 \|p_k\|^2.$$

Therefor, we have

$$-(1-\sigma) g_k^T p_k < \frac{1}{2}\|p_k\|^2 \alpha \Lambda_2 (1+\sigma\ell). \tag{14}$$

Considering (2) and (9) we get $-g_k^T p_k \geq \frac{1}{2} p_k^T B_k p_k$. Combining this inequality and (14) along with Assumption 2 imply that $(1-\sigma)\lambda \|p_k\|^2 < \alpha \|p_k\|^2 \Lambda_2 (1+\sigma\ell)$. Hence the proof is completed. □

Lemma 5. *Let $\{x_k\}$ be the sequence that is generated by Algorithm 1. Then we have*

$$\lim_{k\to\infty} D_k = \lim_{k\to\infty} f(x_k).$$

Proof. We consider the following two cases:

Case 1. $k \in I$. Hence we have

$$\frac{f_{l(k)} - f_{k+1}}{\psi_k(0) - \psi_k(p_k)} \geq \frac{D_k - f_{k+1}}{\psi_k(0) - \psi_k(p_k)} \geq \mu_0.$$

The remainder of the proof is similar to the proof of [1, Theorem 3.2] and thus it is omitted.

Case 2. $k \in J$. Now for $k > \bar{N}$, from (6) we have

$$\begin{aligned} f(x_{l(k)}) &= f(x_{l(k-1)} + \alpha_{l(k-1)} p_{l(k-1)}) \\ &\leq D_{l(k-1)} + \sigma \alpha_{l(k-1)} \left(g_{l(k-1)}^T p_{l(k-1)} - \frac{1}{2} \alpha_{l(k-1)} \ell L_{l(k-1)} \| p_{l(k-1)} \|^2 \right) \\ &\leq f(x_{l(k-1)}) + \sigma \alpha_{l(k-1)} \left(g_{l(k-1)}^T p_{l(k-1)} - \frac{1}{2} \alpha_{l(k-1)} \ell L_{l(k-1)} \| p_{l(k-1)} \|^2 \right). \end{aligned}$$

Since

$$\alpha_{l(k-1)} g_{l(k-1)}^T p_{l(k-1)} < 0, \quad -\frac{1}{2} \alpha_{l(k-1)}^2 \ell L_{lk-1} \| p_{l(k-1)} \|^2 < 0,$$

we have

$$\lim_{k \to \infty} \alpha_{l(k-1)} g_{l(k-1)}^T p_{l(k-1)} = \lim_{k \to \infty} \alpha_{l(k-1)} \| p_{l(k-1)} \| = 0.$$

The remainder of the proof can be found in [13] and thus it is omitted. □

Lemma 6. *Suppose that the sequence $\{x_k\}$ generated by Algorithm 1 is not convergent, i.e., there exists a constant $\varepsilon > 0$ such that*

$$\|g_k\| > \varepsilon, \tag{15}$$

for all k. Therefore, we have $\lim_{k \to \infty} \min \left\{ \Delta_k, \frac{\varepsilon}{\mathscr{L}_k} \right\} = 0$, where $\mathscr{L}_k = 1 + \max\limits_{1 \leq j \leq k} \|B_j\|$.

Proof. We firstly show that there exists a constant $\vartheta > 0$ such that

$$f_{k+1} \leq D_k - \vartheta \min \left\{ \Delta_k, \frac{\varepsilon}{\mathscr{L}_k} \right\}, \quad \forall k \in \mathbb{N}. \tag{16}$$

To this end, we consider two cases:

Case 1. If $k \in I$, then by (5), (9) and (15), we have

$$D_k - f_{k+1} \geq \mu_0 \gamma \|g_k\| \min \left\{ \Delta_k, \frac{\|g_k\|}{\|B_k\|} \right\} \geq \mu_0 \gamma \varepsilon \min \left\{ \Delta_k, \frac{\varepsilon}{\mathscr{L}_k} \right\} = \vartheta_1 \min \left\{ \Delta_k, \frac{\varepsilon}{\mathscr{L}_k} \right\},$$

where $\vartheta_1 = \mu_0 \gamma \varepsilon$.

Case 2. If $k \in J$, then by (6), (10), (15) and Lemma 4, we can write

$$\begin{aligned} f_{k+1} &\leq D_k + \sigma \alpha_k \left(g_k^T p_k - \frac{1}{2} \alpha_k \ell L_k \| p_k \|^2 \right) \leq D_k + \sigma \alpha_k g_k^T p_k \\ &\leq D_k - \frac{\sigma(1-\sigma)\rho\lambda}{\Lambda_2 (1 + \sigma\ell)} \gamma \varepsilon \min \left\{ \Delta_k, \frac{\varepsilon}{\mathscr{L}_k} \right\} = D_k - \vartheta_2 \min \left\{ \Delta_k, \frac{\varepsilon}{\mathscr{L}_k} \right\}, \end{aligned}$$

where $\vartheta_2 = \frac{\sigma(1-\sigma)\rho\lambda}{\Lambda_2(1+\sigma\ell)} \gamma \varepsilon$.

By setting $\vartheta = \min\{\vartheta_1, \vartheta_2\}$, we can see that (16) holds for all $k \in \mathbb{N}$. Therefore, Lemma 5 and taking a limit completes the proof. □

Lemma 7 ([15]). *Let $\{x_k\}$ be the sequence generated by Algorithm 1. Suppose that all conditions of Lemma 6 hold. Then for all sufficiently large k, we have*

$$\Delta_k \geq \frac{\varepsilon}{\mathscr{L}_k}. \tag{17}$$

Theorem 1. *For the sequence $\{x_k\}$ generated by Algorithm 1 we have*

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

Proof. By contrary, assume that there exists a constant $\varepsilon > 0$ such that (15) holds. Thus, there exists an integer $\bar{k}$ such that, for $k \geq \bar{k}$, (17) holds. This fact together with (16), show that

$$f_{l(k)} \geq D_k \geq f_{k+1} + \vartheta \min\left\{\Delta_k, \frac{\varepsilon}{\mathscr{L}_k}\right\} = f_{k+1} + \vartheta\varepsilon\frac{1}{\mathscr{L}_k}.$$

From Lemma 1, we can write

$$f_{l(k+1)} = \max_{0\leq j\leq n_{k+1}}\{f_{k-j+1}\} \leq \max_{0\leq j\leq n_k+1}\{f_{k-j+1}\} = \max\{f_{k+1}, f_{l(k)}\} = f_{l(k)},$$

thus, for $s = 0, 1, ..., \bar{N}$, we have $f_{l(k)} \geq ... \geq f_{l(k+s)} \geq f_{k+s+1} + \vartheta\varepsilon\frac{1}{\mathscr{L}_{k+s}}$. The remainder of the proof is similar to [2, Theorem 3.5] and thus it is omitted. □

Theorem 2. *Suppose that the sequence $\{x_k\}$ generated by Algorithm 1 converges to x^*, $\nabla^2 f(x^*)$ is positive definite, $\|B_k^{-1}g_k\| \leq \delta_k$ and $p_k = -B_k^{-1}g_k$. If the following condition holds*

$$\lim_{k\to\infty} \frac{\|(\nabla^2 f(x^*) - B_k)p_k\|}{\|p_k\|} = 0,$$

then the sequence $\{x_k\}$ converges to x^* superlinearly. Furthermore, the convergence is quadratically whenever $B_k = \nabla^2 f(x_k)$.

Proof. The proof can be found in [2, 15, 20] and thus it is omitted. □

4 Numerical experiments

In this section, we aim to investigate the numerical experiments of Algorithm 1, to demonstrate its performance. Our experiments are performed on a set of test functions with dimensions ranging from 100 to 6000 from Andrei [3] listed in Table 1. The practical implementation of the proposed algorithm is compared with the following algorithms: 1- Non-monotone adaptive TR algorithm (NATR) [16]. 2- A modified non-monotone TR line search algorithm (MNMTRLS) [20]. 3- New non-monotone TR algorithm with new inexact LS (NTR) [17].

Table 1: List of test function.

Problem Name	Dimension	Problem Name	Dimension
Generalized Rosenbrock	100,500	ARGLINC (CUTE)	100
Perturbed Quadratic	100,500	BDEXP (CUTE)	100,500,1000,3000
Diagonal 4	100,500,1000,3000,6000	HARKERP2 (CUTE)	100
Extended BD1	100,500,1000,3000,6000	Extended DENSCHNB (CUTE)	100,500,1000,3000,6000
Perturbed Quadratic	100,500	Extended DENSCHNF (CUTE)	100,500,1000,3000
Fletchcr (CUTE)	100,500	HIMELH (CUTE)	100,500,1000,3000,6000
NONDQUAR (CUTEr)	100,500	Extended Beal	100,500,1000,3000
DQDRTIC (CUTE)	100,500,1000,3000	Extended Penalty	100,500,1000
Almost Perturbed	100,500	Raydan 2	100,500,1000,3000,6000
Purterbed Tridiagonal	100,500	Diagonal 1	100
LIARWHD (CUTE)	100,500,1000,3000	Diagonal 2	100,500
POWER (CUTE)	100,500	Diagonal 3	100,500
CUBE (CUTE)	100	Hager	100
NONSCOMP (CUTE)	100,500	Genaralized Tridiagonal 1	100
Vardim (CUTE)	100,500	Extended Tridiagonal 1	100,500,1000,3000
QUARTC (CUTE)	100,500,1000,3000	Extended TET	100,500,1000,3000,6000
LIARWHD (CUTE)	100,500,1000,3000	Generalized Tridiagonal 2	100
DIXON3DQ (CUTE)	100,500,1000	Diagonal 5	100,500
HIMMELBG (CUTE)	100,500,1000,3000	Extended Himmelblau	100,500
Partial Perturbed	100,500	Generalized PSC1	100,500
DIXMAANA	300,1500,3000	Extended PSC1	100,500,1000,3000
DIXMAANB	300,1500,3000	Extended Powell	100,500
DIXMAANC	300,1500,3000	Extended BD1	100,500,1000,3000,6000
DIXMAAND	300,1500,3000	Extended Maratos	100,500
DIXMAANE	300,1500	Quadratic QF1	100,500
DIXMAANF	300,1500	Extended quadratic penalty QP1	100,500,1000,3000
DIXMAANG	300,1500	Extended quadratic penalty QP2	100,500,1000,3000
DIXMAANH	300,1500	Quadratic	100,500
DIXMAANI	300	Extended quadratic	100,500,1000,3000,6000
DIXMAANJ	300	Extended Tridiagonal 2	100,500,1000,3000
DIXMAANK	300	Broyden Tridiagonal	100,500
DIXMAANL	300	Diagonal 7 (CUTE)	100,500,1000,3000,6000
BDQRTIC (CUTE)	100,500	Diagonal 8 (CUTE)	100,500,1000,3000,6000
TRIDIA (CUTE)	100,500	Full Hessian FH3	100,500,1000,3000,6000
ARWHEAD (CUTE)	100,500,1000,3000	SINCOS	100,500,1000,3000
BG2 (CUTE)	100,500,1000,3000	Diagonal 9	100,500
ENGVAL1 (CUTE)	100	BIGGSB1 (CUTE)	100,500
EDENSCH (CUTE)	100,500		

We performed numerical calculations in MATLAB R2014a (8.3.0.532) programming environment with a long precision format. The codes were run on a PC processor (Intel (R) Celeron (R) CPU N3350 2.4 GHZ) with 4 GB RAM. The stopping criteria are that the number of iterations exceeds 5000 or

$$\|g(x_k)\| \leq \varepsilon = 10^{-5}.$$

For NTRLS algorithm, we consider $\mu_0 = 0.1,\ c_1 = 0.25,\ c_2 = 2,\ \Delta_0 = 10,\ \bar{N} = 15,\ \bar{D} = 6,\ \nu = 10,\ \rho =$

0.5, $\alpha_0 = 0.3$, $\sigma = 0.001$ and $L_0 = 0.5$. In NTR and MNMTRLS algorithms, we set $\eta_0 = 0.5$ and used the following update formula to calculate η_k:

$$\eta_k = \begin{cases} \frac{\eta_0}{2}, & k = 1, \\ \frac{\eta_{k-1}+\eta_{k-2}}{2}, & k \geq 2. \end{cases}$$

In the case of non-identical parameters, NATR, NTR, and MNMTRLS algorithms have been run with the same parameter values specified in [16], [17], and [20] respectively. In all four algorithms, we set $B_0 = I$ where I is the identity matrix and the matrix B_k is updated by the following BFGS formula:

$$B_{k+1}=B_k-\frac{B_k p_k p_k^T B_k}{p_k^T B_k p_k}+\frac{y_k y_k^T}{y_k^T p_k},$$

where $y_k = g_{k+1} - g_k$ and $p_k = x_{k+1} - x_k$. We let $B_{k+1} = B_k$ as $p_k^T y_k \leq 0$. The TR subproblems are solved by the Steihaug scheme [21]. The numerical results of the mentioned algorithms are presented in Tables 2 and 3, in which N_i, N_f and $CPUtime$ respectively represent the number of iterates, number of function evaluations and running time. In addition, (-) indicates that the number of iterations exceeds 5000.

With a glance at Tables 2 and 3 it can be seen that the NATR, NTR, and MNMTRLS algorithms have failed in some cases, while the NTRLS algorithm has solved all the test functions.

Dolan and More [9] proposed a new method to compare the performance of iterative algorithms with a statistical process by displaying performance profiles. Therefore, in order to have a better comparison, we use this technique to visually compare the efficiency of four algorithms according to the numerical results in Tables 2 and 3.

In Fig.s 1, 2 and 3, the efficiency comparisons were drawn using the Dolan and More performance profile [9] on the number of iterations, the number of function evaluations, and the running time, respectively. It is worth noting that in all mentioned algorithms the number of gradient evaluations is equal to the number of iterations. From Fig. 1 we can see that for nearly 65% of the test problems, the NTRLS algorithm is the best solver and in Fig. 2 we observe that NTRLS obtains the most wins in approximately 58% of all the test problems.

From Fig. 3, we observe that in more than 40% of cases, the new algorithm is faster than the other algorithms. Another important factor of these three figures is that the graph of the NTRLS algorithm grows up faster than the others, which means in the cases that this algorithm does not get the best result, it performs closer to the performance index of the best algorithm. Therefore, we can conclude that the new algorithm is more efficient and robust compared to the others mentioned algorithms.

5 Conclusions

Considering the advantages of the non-monotone technique, to avoid resolving the TR subproblem, we have presented a new non-monotone LS method in the TR framework, which prevents the sudden increase of objective function values in the non-monotone TR method. In other words, we have used the Armijo-type LS method in the direction of the failed trial step to create a new point. The global and superlinear properties have been proven under appropriate conditions. Finally, the proposed algorithm has been tested on 75 test functions. Comparative numerical experiments have clearly shown the efficiency and robustness of the proposed algorithm using the Dolan-More performance profile.

Table 2: Numerical results

N_i/N_f				*CPUtime*				N_i/N_f				*CPUtime*			
NTRLS	NATR	NTR	MNMTRLS	NTRLS	NATR	NTR	MNMTRLS	NTRLS	NATR	NTR	MNMTRLS	NTRLS	NATR	NTR	MNMTRLS
896/1132	994/1266	972/1177	-	9.2344	8.5781	9.7188	-	22/24	17/19	21/22	21/22	0.0625	0.0625	0.0625	0.15625
157/225	154/241	195/263	744/791	0.70313	0.75	0.84375	3.1094	6/10	8/10	8/12	8/12	0.03125	0.03125	0.03125	0.03125
3/4	3/5	3/4	3/4	0.015625	0	0.03125	0	61/95	54/102	200/270	216/289	0.21875	0.25	0.73438	1.0938
10/14	9/11	14/16	10/13	0.046875	0.03125	0.03125	0.0625	5/6	5/6	5/6	5/6	0.015625	0.03125	0	0.03125
22/25	22/24	22/25	32/57	0.17188	0.15625	0.15625	0.25	8/12	13/18	10/13	11/15	0.015625	0.0625	0.03125	0.0625
500/683	746/1036	506/665	-	2.5781	4.5156	2.5625	-	201/207	83/86	244/251	429/463	3.0781	0.57813	4.9219	16.656
626/633	633/642	678/683	622/627	32.016	33.891	38.469	29.656	19/20	11/13	19/20	34/56	0.078125	0.03125	0.046875	0.15625
14/23	57/90	16/18	25/51	0.078125	0.3125	0.0625	0.09375	73/108	249/346	322/375	394/447	0.67188	1.7656	11.391	8.9844
159/226	159/248	198/267	696/736	0.67188	0.6875	0.75	2.3906	10/14	9/11	14/16	10/13	0.046875	0.03125	0.0625	0.03125
177/252	170/267	187/252	921/977	0.64063	0.78125	0.79688	3.3125	649/814	664/835	954/1141	-	10.313	12.609	15.984	-
20/24	15/22	17/20	31/57	0.078125	0.0625	0.046875	0.125	289/392	397/550	219/293	408/447	1.8281	2.7031	1.0156	1.75
628/851	909/1249	630/815	-	3.5	5.5313	4.1563	-	23/24	11/14	22/25	22/24	0.078125	0.0625	0.046875	0.0625
3506/4687	2534/3265	2089/2675	325/441	38.719	28	20.469	3.2813	30/40	30/41	23/33	26/38	0.10938	0.10938	0.09375	0.09375
132/188	705/833	1041/1334	264/396	0.5625	8.2344	18.266	2.7656	378/516	545/757	204/275	1032/1077	1.8281	3.1406	0.84375	3.8906
34/35	34/36	34/35	35/47	0.09375	0.125	0.125	0.125	4/9	4/10	4/9	5/10	0.03125	0	0.03125	0
1/2	14/15	1/2	1/2	0	0.03125	0	0	57/62	58/60	58/62	59/74	0.4375	0.45313	0.45313	0.45313
20/24	15/22	19/22	39/65	0.078125	0.0625	0.0625	0.15625	91/135	44/88	188/254	262/305	0.375	0.21875	0.78125	1.1875
125/151	139/166	123/135	126/131	1.7031	1.4531	1.5625	1.5938	7/8	6/7	7/8	7/8	0.015625	0.03125	0.03125	0.03125
17/18	17/18	17/18	11/23	0.078125	0.0625	0.0625	0.03125	4/7	6/7	4/7	4/7	0	0.03125	0.03125	0
79/118	78/143	210/283	316/364	0.375	0.35938	0.92188	1.4688	3/4	4/6	3/4	3/4	0	0.015625	0.03125	0
7/8	8/10	7/8	7/8	0.23438	0.28125	0.21875	0.21875	19/20	11/13	19/20	34/56	0.046875	0.0625	0.0625	0.17188
9/10	9/10	9/10	9/10	0.3125	0.28125	0.26563	0.29688	293/392	415/543	161/216	-	2.5469	3.2969	0.75	-
8/9	11/12	8/9	8/9	0.26563	0.375	0.28125	0.28125	120/145	135/167	125/133	123/128	1.6406	1.4688	1.6875	1.6719
8/9	10/11	8/9	8/9	0.28125	0.34375	0.25	0.28125	4252/5561	-	2722/3572	-	791.64	-	247.09	-
80/81	81/82	80/81	80/81	5.6094	5.5156	5.5	5.4531	815/1115	807/1262	703/930	4576/4693	71.859	75.5	62.391	408.08
75/76	75/76	75/76	75/76	5.0781	5.0781	4.9844	5.0313	3/4	5/7	3/4	4/5	0.25	0.42188	0.25	0.45313
72/73	72/73	72/73	72/73	4.7344	4.7813	4.8125	4.6406	10/13	11/13	13/15	15/40	0.75	0.82813	1.0313	0.9375
65/66	66/67	65/66	65/66	4.125	4	3.9688	4	107/149	136/189	604/605	563/564	25.422	31.984	121.11	105.5
569/570	568/569	569/570	575/596	152.33	147.38	148.97	155.2	2609/3529	3727/5113	2696/3524	-	322.42	449.38	317	-
434/435	434/435	434/435	434/435	88.641	86.172	86.328	89.859	1195/1201	1177/1186	1169/1175	1256/1261	1824.4	1769.4	2098.3	2086.3
420/421	420/421	420/421	420/421	85.063	84.531	82.094	80.703	18/28	54/87	22/27	20/22	1.8906	5.3125	2.2344	2.0469
397/398	394/395	397/398	397/398	74.484	73.844	75.734	75.922	813/1112	816/1254	783/1037	4598/4713	69.828	71.172	64.328	942.72
332/452	501/691	382/502	1567/1673	1.8906	2.7813	2.2813	6.3906	867/1186	806/1252	791/1048	-	75.984	72.453	66.031	-
455/624	647/950	460/595	-	2.1875	3.6406	2.375	-	16/18	34/48	17/18	24/28	1.4063	3.4688	1.4531	2.1406
7/12	8/14	8/12	7/12	0.015625	0.03125	0.03125	0.03125	4539/6146	-	4528/5899	-	534.03	-	605.44	-
11/14	50/56	45/51	13/16	0.03125	0.23438	0.1875	0.0625	142/203	2348/3108	2571/3339	-	12.813	506.41	674.73	-
205/264	291/353	216/267	1250/1358	2.2031	2.875	2.0781	21.719	44/45	45/47	44/45	44/45	3.7813	3.75	3.6406	3.6719
207/274	302/408	210/271	1023/1110	1.8906	2.5156	1.75	5.4375	19/20	15/16	19/20	21/44	1.5781	1.25	1.5625	1.8438
2/3	3/5	2/3	2/3	0.015625	0.03125	0	0.09375	17/19	29/38	17/18	24/28	1.5625	2.7813	1.6094	2.1094
17/18	19/20	17/18	19/42	0.078125	0.046875	0.0625	19.047	601/644	602/641	594/603	591/595	281.89	253.38	255.36	271.45
250/278	506/544	259/276	278/290	19.781	13.719	17.156	0.03125	19/20	43/44	19/20	19/31	1.5	3.2188	1.6719	1.6094
8/9	7/9	8/9	8/9	0.015625	0	0.03125	0.03125	368/510	299/489	358/475	1486/1724	33.156	29.641	32.875	280.17
11/15	14/25	11/15	13/28	0.0625	0.09375	0.03125	0	9/10	8/10	9/10	9/11	13.984	12.359	14.078	14.141
6/9	6/8	6/9	7/10	0.03125	0.03125	0.03125	0	9/10	9/10	9/10	10/33	14.438	14.266	14.672	16.063
14/23	26/42	44/56	17/19	0.046875	0.10938	0.26563	0.09375	9/10	12/13	9/10	13/36	14.891	19.625	14.766	21.25
37/40	58/59	52/55	62/76	0.15625	0.21875	0.20313	0.25	10/11	11/12	10/11	14/37	16.672	18.156	16.234	22.5
1/2	6/7	1/2	1/2	0.03125	0.015625	0	0	170/171	171/172	170/171	170/171	627.73	625.66	628.34	620.23
283/378	414/545	222/297	305/380	1.9688	3.0469	1	1.9531	137/138	137/138	137/138	137/138	454.52	452.63	450.06	451.09
79/80	78/79	79/80	80/103	0.64063	0.625	0.59375	0.57813	134/135	134/135	134/135	134/135	432.23	430.06	435.55	430.02
293/396	403/554	184/249	1030/1101	1.75	2.7656	0.85938	3.6094	130/131	130/131	130/131	130/131	414.19	414.88	412.42	412.75
174/188	205/231	177/197	2108/2336	1.5781	2.0469	1.7031	12.703	858/1165	2114/2973	-	-	85.156	226.81	-	-
258/345	341/462	267/345	323/425	2.0781	2.7813	2.0781	3.3594	2969/4030	3742/5587	2781/3639	-	275.84	389.17	258.16	-

Table 3: Numerical results

N_i/N_f				*CPUtime*				N_i/N_f				*CPUtime*			
NTRLS	NATR	NTR	MNMTRLS	NTRLS	NATR	NTR	MNMTRLS	NTRLS	NATR	NTR	MNMTRLS	NTRLS	NATR	NTR	MNMTRLS
7/12	18/135	8/12	9/21	0.64063	1.7031	0.6875	1.0625	7/8	6/7	7/8	10/33	3.2031	2.8438	3.2656	4.5938
12/15	22/27	128/165	101/111	1.1406	1.9688	15.703	10.266	23/24	25/27	23/24	23/24	11.281	12.313	11.203	11.172
824/1119	1250/1713	150/202	4470/4818	164.58	206.58	21.641	1625.9	9/12	9/11	9/12	8/11	4.2031	4.3906	4.3438	3.7344
20/21	21/22	20/21	22/45	1.9219	2.1563	1.9219	2.0625	16/17	21/22	16/17	20/43	7.6875	9.6875	7.5625	9.4844
9/10	7/10	9/10	12/25	0.75	0.625	0.79688	1.0156	11/12	12/14	11/12	11/13	4.8125	4.7031	4.8438	4.7344
15/21	14/21	42/52	12/15	1.2656	1.3438	4.5313	1.0313	19/21	13/16	21/23	28/52	10.547	6.3594	9.9219	13.047
6/8	6/8	6/7	6/7	0.64063	0.53125	0.51563	0.53125	92/115	20/25	66/84	84/120	41.922	9.0938	28.75	35.016
14/22	35/55	24/26	16/27	1.3125	3.5469	2.5469	1.4219	4/8	4/9	4/8	5/9	1.9063	1.7969	2	2.2656
101/111	253/255	82/96	126/139	13.375	20.25	9.1406	11.203	80/83	60/63	80/83	75/79	54.172	40.734	53.703	49.391
162/163	169/170	162/163	164/176	29.359	29.438	29.25	30.453	7/8	6/7	7/8	10/33	3.1406	2.9063	3.2969	4.5625
893/1231	2212/3073	-	-	69.188	178.78	-	-	4/5	6/7	4/5	4/5	1.75	2.9688	1.875	2.2188
19/21	23/25	19/21	24/37	1.9063	2.1719	1.6875	2.1406	5/6	4/5	5/6	7/30	2.375	1.7656	2.4531	3.1875
8/11	7/9	8/11	10/14	0.625	0.54688	0.60938	0.8125	16/17	21/22	16/17	20/43	8.2031	10.672	8.1094	10.438
5/6	6/7	5/6	8/31	0.51563	0.46875	0.40625	0.65625	4/5	8/15	4/5	6/29	40.172	78.766	39.125	58.781
11/14	10/14	13/15	13/16	0.82813	0.84375	1.0156	1.1719	12/16	12/13	12/15	14/30	99.219	100.67	99.625	118.13
186/198	99/101	636/669	261/294	74.109	14.563	1129.9	74.375	20/31	57/88	26/29	36/49	207.17	574.17	269.73	367.5
15/16	13/14	15/16	19/42	1.2344	1.0625	1.2188	1.5781	33/45	62/91	36/47	208/242	350.88	633.56	383.64	2222.4
606/826	1060/1469	927/1204	968/1198	138.88	154.75	230.84	402.89	21/22	19/20	21/22	23/46	206.58	187.41	208.75	228.98
10/13	11/13	13/15	15/40	0.82813	0.9375	1.0625	0.90625	32/43	57/84	35/43	53/57	337.2	577.25	367.63	577.19
1249/1702	1475/2043	1000/1312	-	173.06	178.48	98.984	-	22/23	53/54	22/23	22/24	218.73	520.27	221.38	220.8
839/1147	1516/2217	725/958	1892/1954	71.547	136.06	60.625	378.7	8/13	15/25	9/13	10/21	80.125	151.06	88.672	99.359
21/23	74/78	20/22	24/47	1.8125	2.0781	1.7031	2.0313	16/19	14/19	13/15	154/174	165.78	141.3	131.97	1613.1
197/242	28/35	140/170	235/298	28.125	2.375	18.891	22.141	22/23	26/27	22/23	23/35	224.67	260.94	222.77	238.14
1126/1536	1928/2956	856/1135	-	94.453	173.52	70.016	-	11/12	11/13	11/12	13/26	109.22	106.45	112.5	129.23
4/9	4/10	4/9	5/10	0.3125	0.29688	0.3125	0.375	18/23	11/13	33/36	36/50	184.3	111.69	345.33	359.05
72/76	72/74	72/76	67/82	11.453	10.484	11.078	10.141	6/7	7/8	6/7	8/9	58.844	69.203	59.656	81.063
424/585	2026/2810	505/669	497/540	36.5	209.52	41.781	43.969	18/29	21/41	23/24	33/47	184.8	214.89	237.31	344.05
6/9	6/7	7/8	8/22	0.60938	0.53125	0.54688	0.67188	7/8	6/7	7/8	7/8	70.281	59.609	69.109	69.578
6/7	6/7	6/7	6/7	0.46875	0.46875	0.46875	0.46875	23/24	10/11	23/24	25/27	233.67	100.97	233.02	251.27
4/5	4/5	4/5	7/30	0.34375	0.3125	0.3125	0.5625	7/8	9/11	7/8	8/9	70.422	90.031	69.688	80.188
15/16	13/14	15/16	19/42	1.3125	1.0781	1.25	1.5469	19/21	29/30	21/23	48/59	191.8	288.97	210.86	521.39
1015/1385	2038/2821	-	-	92.578	195.66	-	-	12/16	12/13	12/15	14/30	101.06	101.58	99.516	120.3
589/614	602/651	591/600	589/593	274.7	234.5	251.64	260.61	29/32	19/20	27/30	30/54	294.02	194.83	273.02	300.69
4/5	4/5	4/5	6/29	1.9844	1.9063	1.7813	2.6875	85/94	150/160	68/84	73/108	783.66	369.16	630.41	683.88
11/12	12/14	11/12	11/13	5	4.8438	4.625	4.625	4/8	4/9	4/8	4/8	44.391	40.25	39.813	39.594
19/30	56/92	20/25	52/56	9.5	27.766	10.203	26.938	65/69	61/64	72/75	72/75	757.63	712.66	828.64	813.33
18/20	42/60	18/19	20/21	8.7188	21.063	8.7031	9.9844	7/8	6/7	7/8	9/23	70.203	59.313	70.266	89.281
20/21	11/12	20/21	22/45	9.2656	5	9.0625	10.109	6/7	6/7	6/7	6/7	59.906	59.516	60.016	59.875
17/19	37/49	17/18	22/23	8.375	18.078	8.0469	10.719	5/6	5/6	5/6	4/5	50	48.906	49.422	40.203
20/21	50/51	20/21	21/33	9.4063	22.484	9.2813	9.8125	19/21	29/30	21/23	48/59	191.64	284.17	213.83	523.11
9/10	8/10	9/10	11/34	95.297	85.109	95.781	116.13	5/6	28/40	5/6	4/5	386.48	2164.4	383.69	307.34
8/9	9/10	8/9	10/33	84.984	95.281	84.484	104.7	10/14	13/14	17/18	10/12	620.84	847.2	1178	702.67
9/10	12/13	9/10	12/35	95.922	128.34	95.781	127.09	10/12	16/18	10/12	9/10	773.69	1225.3	767.05	692.69
11/12	11/12	11/12	12/35	117.13	119.56	121.36	131.81	7/8	6/7	7/8	10/23	532.19	451.31	526.47	755.36
8/13	9/15	9/13	9/20	4.4063	4.4531	4.2344	4.5469	8/9	9/10	8/9	7/8	600.16	673.34	599.09	524.83
16/19	14/19	16/21	112/126	7.5938	6.8281	7.0156	57.297	7/8	8/10	7/8	8/9	529.81	602.73	525.86	599.95
20/21	21/22	20/21	23/46	10.094	10.5	9.8125	11.422	10/14	13/14	17/18	10/12	603.63	824.23	1138.6	674.58
8/9	7/8	8/9	13/26	4	3.2656	3.7969	6.0156	4/8	4/8	4/8	5/10	298.3	299.92	298.36	373
20/26	13/25	137/182	27/52	9.3594	6.2188	72.875	13.25	8/9	6/7	8/9	7/8	595.94	466.61	623.11	563.36
7/9	6/8	6/7	6/7	3.4844	2.7813	2.8125	2.7656	6/7	6/7	6/7	9/32	457.98	453.58	450.59	676.47
15/23	23/40	34/42	20/21	7.4844	11.391	18.484	10.016	6/7	8/9	6/7	5/6	455.55	600.97	451.59	376.13
82/86	665/667	95/111	291/303	41.5	300.64	52.313	135.7								

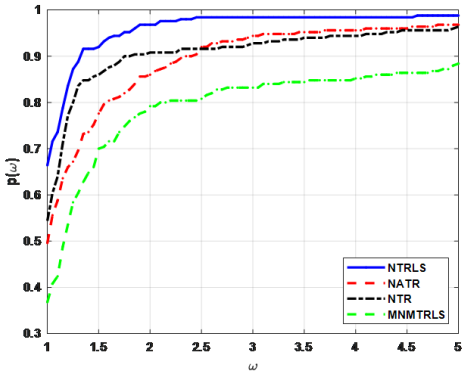

Figure 1: Performance profiles for number of iterations

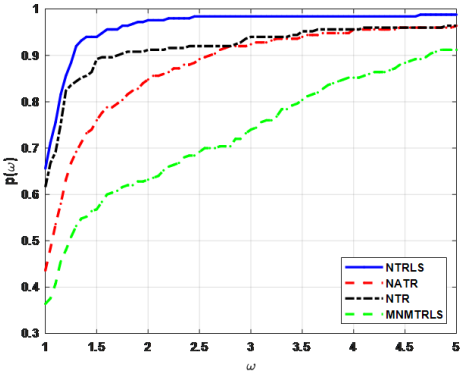

Figure 2: Performance profiles for number of function evaluations

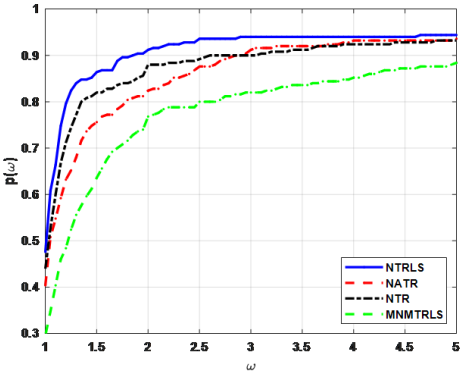

Figure 3: Performance profiles for the running time

References

[1] M. Ahookhosh, K. Amini, *A non-monotone trust region method with adaptive radius for unconstrained optimization*, Comput. Math. Appl. **60** (2010) 411–422.

[2] M. Ahookhoosh, K. Amini, M.R. Peyghami, *A non-monotone trust region lin search method for large scale unconstrained optimization*, Appl. Math. Model. **36** (2012) 478–487.

[3] N. Andrei, *An unconstrained optimization test functions collection*, Adv. Model. Optim. **10** (2008) 147–161.

[4] L. Armijo, *Minimization of functions having Lipschitz continuous first partial derivatives*, Pac. J. Math. **16** (1966) 1–3.

[5] R.M. Chamberlain, M.J.D. Powell, C. Lemarechal, H.C. Pedersen, *The watchdog technique for forcing convergence in algorithm for constrained optimization*, Math. Program. Stud. **16** (1982) 1–17.

[6] A.R. Conn, N.I.M. Gould, P.L. Toint, *Trust Region Methods*, MPS/SIAM Series on Optimization, SIAM, Philadelphia, 2000.

[7] N.Y. Deng, Y. Xiao, F.J. Zhou,*Non-monotonic trust region algorithm*, J. Optim. Theory. Appl. **76** (1993) 259–285.

[8] X. Ding, Q. Qu, X. Wang, *A modified filter non-monotone adaptive retrospective trust region method*, PLOS ONE. **16** (2021) e0253016.

[9] E.D. Dolan, J.J. More, *Benchmarking optimization software with performance profiles*, Math. Program. **91** (2002) 201–213.

[10] J.H. Fu, W.Y. Sun, *Non-monotone adaptive trust region method for unconstrained optimization problems*, Appl. Math. Comput. **163** (2005) 489–504.

[11] E.M. Gertz, *Combination Trust Region Line Search Methods for Unconstrained Optimization*, University of California, San Diego, 1999.

[12] A.A. Goldstein, *On steepest descent*, SIAM J. Contro. **3** (1965) 147–151.

[13] L. Grippo, F. Lampariello, S. Lucidi, *A non-monotone line search technique for Newtons method*, SIAM J. Numer. Anal. **23** (1986) 707–716.

[14] L. Grippo, F. Lampariello, S. Lucidi, *A truncated Newton method with non-monotone line search for unconstrained optimization*, J. Optim. Theory. Appl. **60** (1989) 401–419.

[15] N.Z. Gu, J.T. Mo, *Incorporating non-monotone strategies into the trust region method for unconstrained optimization*, Appl. Math. Comput. **55** (2008) 2158–2172.

[16] A. Kamandi, K. Amini, *A new non-monotone adaptive trust region algorithm*, Appl. Math. **67** (2021) 233–250.

[17] J. Liu, C. Ma, *A non-monotone trust region method with new inexact line search for unconstrained optimization*, Numer. Algorithms **64** (2013) 1–20.

[18] J. Nocedal, S.J. Wright, *Numerical Optimization*, Springer, NewYork, 2006.

[19] J. Nocedal, Y.X. Yuan, *Combining trust region and line search techniques*, In: Yuan, Y. (ed.) Advances in Nonlinear Programming, pp. 153–175. Kluwer Academic Publishers, Dordrecht, 1998.

[20] S. Rezaee, S. Babaie-Kafaki, *A modified non-monotone trust region line search method*, J. Appl. Math. Comput. **57** (2018) 421–436.

[21] T. Steihaug, *The conjugate gradient method and trust regions in large scale optimization*, SIAM J. Numer. Anal. **20** (1983) 626–637.

[22] P.L. Toint, *An assessment of non-monotone line search techniques for unconstrained optimization*, SIAM J. Sci. Comput. **17** (1996) 725–739.

[23] Z. Wan, S. Huang, X.D. Zheng, *New cautious BFGS algorithm based on modified Armijotype line search*, J. Inequal. Appl. **2012** (2012) 241.

[24] P. Wolfe, *Convergence conditions for ascent methods*, SIAM. Rev. **11** (1986) 226–235.

[25] X. Xiao, E.K.W. Chu, *Non-monotone trust region methods*, Technical Report 95/17. Monash University. Clayton, Australia, 1995.

[26] H. Zhang, W.W. Hager, *A non-monotone line search technique and its application to unconstrained optimization*, SIAM J. Optim. **14** (2004) 1043–1056.

[27] F. Zhou, Y. Xiao, *A class of non-monotone stabilization trust region methods*, Computing **53** (1994) 119–136.