AIMS Mathematics, 9(11): 31790–31805.
DOI: 10.3934/math.20241528
Received: 13 September 2024
Revised: 17 October 2024
Accepted: 31 October 2024
Published: 08 November 2024

https://www.aimspress.com/journal/Math

Research article

A novel nonmonotone trust region method based on the Metropolis criterion for solving unconstrained optimization

Yiting Zhang, Chongyang He, Wanting Yuan and Mingyuan Cao*

School of Mathematics and Statistics, Beihua University, Jilin 132013, China

* **Correspondence:** Email: cmy0918@beihua.edu.cn.

Abstract: In this paper, we propose a novel nonmonotone trust region method that incorporates the Metropolis criterion to construct a new function sequence. This sequence is used to update both the trust region ratio and the iteration criterion, increasing the likelihood of accepting the current trial step and introducing randomness into the iteration process. When the current trial step is not accepted, we introduce an improved nonmonotone line search technique to continue the iteration. This approach significantly reduces the number of subproblems that need to be solved, thereby saving computational resources. The stochastic nonmonotone technique helps the algorithm avoid being trapped in the local optima, and a global convergence is guaranteed under certain conditions. Numerical experiments demonstrate that the algorithm can be more effectively applied to a broader range of problems.

Keywords: optimization; trust region method; line search; metropolis criterion; nonmonotone
Mathematics Subject Classification: 49M37, 65K05, 90C30

1. Introduction

Unconstrained optimization is one of the fundamental problems in the optimization theory [4, 26]. It has broad applications across various fields such as engineering design, financial investments, signal processing, and so on [6, 18, 21]. In engineering design, unconstrained optimization is used to determine the most efficient design parameters for systems or products, which may either include optimizing the dimensions of mechanical components or minimizing material usage while maintaining the structural integrity. In financial investments, unconstrained optimization techniques enable investors to identify the optimal portfolio allocation, thereby balancing the risk and the return without being confined by predefined constraints. This helps investors maximize the expected returns while minimizing the potential losses. In signal processing, unconstrained optimization is employed in tasks such as signal denoising, image quality enhancement, and performance optimization of communication systems. By either minimizing error functions or maximizing the signal-to-noise

ratio, unconstrained optimization algorithms can extract meaningful information from noisy data. Therefore, the discussion of methods for solving unconstrained optimization problems becomes very important.

The trust region method is one of the most commonly used iterative algorithms to solve unconstrained optimization problems [14, 27]. The fundamental concept behind the trust region method lies in constructing a surrogate model that approximates the objective function within the neighborhood of the current iteration point. This surrogate model captures the local behavior of the original objective function, thus allowing the algorithm to compute the optimal step size or the trial step within the predefined trust region.

In most practical implementations, a quadratic model is typically chosen as the approximation function due to its mathematical convenience and the availability of efficient solvers. However, the effectiveness of the quadratic model heavily depends on the proper tuning of its parameters. Researchers have proposed various techniques to solve trust region subproblems, including exact solutions, dog-leg methods, and conjugate gradient methods [1, 5, 33].

However, trust region subproblems are often difficult to solve, particularly for complex objective functions and high-dimensional problems, thus leading to increased computational costs. The challenges typically arise from the non-convexity and multimodal nature of the surrogate function within the trust region, requiring frequent subproblem solutions during the iterative process. To address this, researchers have explored methods to improve the efficiency of solving these subproblems by developing more efficient solvers, employing advanced optimization techniques, and exploring alternative approximation functions [8, 22, 23].

Overall, trust region methods remain powerful tools for unconstrained optimization, and further advancements in subproblem solvers could enhance their performance and applicability to a wider range of problems. To handle situations where the trial step is not accepted, trust region methods combined with line search techniques have been proposed [11, 25]. When the trial step is rejected, these algorithms use the trial step as the search direction and apply a line search technique to determine the corresponding step size. However, these methods require the objective function to monotonically decrease at each iteration, which has some impact on the convergence rate of the algorithm. With the introduction of nonmonotone techniques [12], a new class of nonmonotone trust region algorithms was developed [7], which allowed the sequence of the objective function values to be nonmonotone. Numerical results show that these methods outperform traditional trust region methods. Further research has been conducted in this area [10, 28, 29].

However, in some cases, the class of aforementioned nonmonotone techniques that used a maximum of recent function values may ignore some better function values. To address this, scholars have proposed improved nonmonotone strategies. Zhang and Hager [34] introduced a new nonmonotone line search algorithm that replaced the maximum function value with the average of recent function values. Gu and Mo [13] developed a simplified nonmonotone strategy that was applied within the trust region framework, which avoided the complex parameterization process of Zhang and Hager [34]. Zhou et al. [35] proposed a nonmonotone trust region method based on a simple model, which performed well on large-scale problems. Improved nonmonotone techniques have also been applied to solve other problems, such as the nonmonotone BFGS (Broyden, Fletcher, Goldfarb and Shanno) algorithm proposed by Wan et al. [31] for smooth nonlinear equations, where the nonmonotone parameters were updated using known information of the nonlinear equation. Some

scholars have developed new nonmonotone line search rules that were applied to unconstrained and box-constrained optimization problems [16, 19]. In contrast to previous work, we study a stochastic nonmonotone technique that allows the algorithm to explore a larger solution space during the search process.

To reduce the number of subproblem computations and improve the overall efficiency of the algorithm, we propose a modified trust region method that incorporates recent advances in this field, with the following key innovations and advantages.

Construction of a New Function Sequence: To prevent convergence to the local optima and to improve algorithmic performance, we introduce the idea of simulated annealing to construct a new function sequence. Based on this sequence, a new nonmonotone trust region ratio is defined, which leads to corresponding improvements in the iteration criterion of the trust region framework, thus providing a greater flexibility during the iterative process.

Improved Nonmonotone Strategy: When the current trial step is not accepted, we utilize an improved nonmonotone strategy that leverages the new function sequence to determine the step size. This approach effectively reduces the required number of subproblem solutions, thus enhancing the algorithm's overall efficiency.

Global Convergence and Efficiency: Our algorithm guarantees a global convergence under certain assumptions. Furthermore, numerical experiments demonstrate that the algorithm significantly reduces the number of iterations, improves the efficiency, and delivers a strong performance.

The structure of this paper is as follows: in Section 2, we describe the new nonmonotone trust region algorithm; in Section 3, the global convergence of our algorithm is proved under certain assumptions; in Section 4, the related numerical experimental results are given; and finally, the conclusion of this paper is given in Section 5.

2. New nonmonotone trust region method

Consider the following unconstrained optimization problem:

$$\min_{x\in\mathbb{R}^n} f(x), \tag{2.1}$$

where $f\colon \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable. The difficulty in solving the trust region problem often stems from the need to balance multiple objectives: minimizing the approximate objective function, satisfying the trust region constraint, and potentially incorporating additional requirements such as sparsity or regularization. The trust region method computes the trial step d_k by solving the following subproblem:

$$\min_{d\in\mathbb{R}^n} m_k(d) = g_k^T d + \frac{1}{2} d^T B_k d, \quad \text{s.t. } \|d\| \le \Delta_k, \tag{2.2}$$

where $g_k = \nabla f(x_k)$, B_k is the approximation of the Hessian matrix at x_k, Δ_k is the trust region radius, and $\|\cdot\|$ denotes the Euclidean norm.

As we all know, in trust region method, the trust region ratio is the key to determining whether the algorithm iterates using the current trial step. The following original trust region ratio is monotone:

$$r_k = \frac{Ared}{Pred} = \frac{f_k - f(x_k + d_k)}{m_k(0) - m_k(d_k)}, \tag{2.3}$$

where $f_k = f(x_k)$, *Ared* is the actual reduction, and *Pred* is the predicted reduction. The monotone technique may slow down the rate of convergence, particularly in a narrow curved valley. In order to overcome this disadvantage, the nonmonotone trust region ratio [30] has been proposed:

$$r_k = \frac{f_{lk} - f(x_k + d_k)}{m_k(0) - m_k(d_k)},$$

where $f_{lk} = \max\limits_{0 \le j \le q(k)} f(x_{k-j})$, N is a fixed positive constant, and

$$q(k) = \begin{cases} k, & if \ \ k \le N, \\ \min\{q(k-1)+1, N\}, & if \ \ k > N. \end{cases}$$

However, this method may result in missing some well-performing function values, so some scholars have proposed improved nonmonotone algorithms. Gu and Mo [13] constructed the following trust region ratio in 2008:

$$r_k = \frac{D_k - f(x_k + d_k)}{m_k(0) - m_k(d_k)},$$

where

$$D_k = \begin{cases} f_k, & if \ \ k = 0, \\ \eta D_{k-1} + (1-\eta) f_k, & if \ \ k \ge 1, \end{cases}$$

and $0 < \eta < 1$ or a variable η.

Based on the inspiration from the above ideas, we propose a new nonmonotone trust region algorithm, which allows the algorithm to use the existing favorable information, and also avoids the algorithm from falling into local optimal solutions. To achieve these, we consider introducing the idea of simulated annealing [15, 17, 32] into the trust region method. A new trust region framework is constructed, which includes the improvement of the trust region iteration criterion and the trust region ratio, etc.

First, we construct a new trust region ratio:

$$r_k = \frac{R_k - f(x_k + d_k)}{m_k(0) - m_k(d_k)}, \tag{2.4}$$

where

$$R_k = \begin{cases} f_k, & if \ \ k = 0, \\ (1 - p_{k-1}) R_{k-1} + p_{k-1} f_k, & if \ \ k \ge 1, \end{cases} \tag{2.5}$$

and

$$p_k = \begin{cases} 1, & if \ \ r_k \ge \tau, \\ \exp\left\{-\frac{\tau - r_k}{T_k}\right\}, & otherwise. \end{cases} \tag{2.6}$$

Here, we introduce the above parameter p_k [3, 20] into the R_k, where T_k is the temperature during annealing, τ is a given constant, and r_k is the trust region ratio. It is crucial to control the rate of the annealing cooling process [2], which is usually updated according to the following:

$$T_{k+1} = \beta T_k, \quad where \ \ 0 < \beta < 1.$$

From the definition of p_k, we know that it is dynamically updated by the value of r_k. From this, it can be observed that our trust region ratio r_k can be adaptively adjusted according to the existing information.

Furthermore, considering the relation between r_k and τ in the parameter p_k, we use the parameter p_k to determine whether the current trial step d_k is accepted. For a random number $l_k < 1$, if $p_k \geq l_k$ is satisfied, then the current trial step is accepted. When $r_k \geq \tau$, $p_k = 1$, the current trial step is naturally accepted. Otherwise, the current trial step will be accepted with a certain probability, and the larger p_k is more probable when the current trial step is accepted. If $p_k < l_k$, then the current trial step is rejected. Using our sequence R_k, we propose a new nonmonotone line search:

$$f\left(x_k + \lambda^i d_k\right) \leq R_k + \delta\lambda^i g_k^T d_k,$$

where $0 < \delta < 1$, $0 < \lambda < 1$. According to the line search, the corresponding step size is obtained for the iteration. The parameter i_m is computed by this method, which is iterated through $x_{k+1} = x_k + \lambda^{i_m} d_k$. The nonmonotone line search doesn't require the objective function to be monotonically decreasing, which makes the overall algorithm more flexible and faster.

Combined with the above improvements, we propose the nonmonotone trust region algorithm based on the Metropolis criterion (Algorithm 2.1).

Algorithm 2.1.

Step 0. Given an initial point $x_0 \in \mathbb{R}^n$, $R_0 = f(x_0)$, $\Delta_0 > 0$, $T_0 > 0$, $\varepsilon > 0$, $v > 1$, $0 < \lambda < 1$, $0 < \delta \leq 0.5$, $0 < \eta_1 < \eta_2 < 1$, $0 < \gamma_1 < 1 < \gamma_2$, $0 < \beta < 1$, $0 < \tau \leq \eta_1$,$B_0 = \xi I, \xi > 0$, let $k := 0$.

Step 1. Compute g_k. If $\|g_k\| \leq \varepsilon$, stop.

Step 2. Solve the subproblem (2.2) to obtain d_k.

Step 3. Compute r_k and p_k by (2.4) and (2.6), respectively. Additionally, compute the following:

$$l_k = e^{-v} + \left(e^{-1/v} - e^{-v}\right) \times rand(1).$$

Step 4. If $p_k \geq l_k$, then $x_{k+1} = x_k + d_k$. Otherwise, compute i_m, which is the minimum nonnegative integer i which satisfies the following:

$$f\left(x_k + \lambda^i d_k\right) \leq R_k + \delta\lambda^i g_k^T d_k;$$

then, $x_{k+1} = x_k + \lambda^{i_m} d_k$.

Step 5. Update the trust region radius:

$$\Delta_{k+1} = \begin{cases} \gamma_1\Delta_k, & if \ \ r_k \leq \eta_1, \\ \gamma_2\Delta_k, & if \ \ r_k \geq \eta_2, \\ \Delta_k, & otherwise. \end{cases}$$

Step 6. Compute f_{k+1} and $R_{k+1} = (1 - p_k) R_k + p_k f_{k+1}$, $T_{k+1} = \beta T_k$, and compute B_{k+1} by a quasi-Newton update. Set $k := k + 1$, then go to Step 1.

Remark 2.1. In Step 3, if $r_k \geq \tau$, then the function approximation of the previous iteration is better; then, $p_k = 1$ and $R_{k+1} = f_{k+1}$, which is reduced to the case of the traditional trust region ratio. If $r_k < \tau$, then $0 < p_k < 1$, and R_{k+1} is the convex combination of R_k and f_{k+1}.

3. Convergence analysis

Nonmonotone trust region methods have good global convergence as described in [13, 24]. In this section, we will analyze the convergence of our algorithm, which also inherits the properties of nonmonotone trust region methods. First, the following assumptions are made.

Assumption 3.1. *Let* $L = \{x \in \mathbb{R}^n \,|f(x) \leq f(x_0)\} \subset \Omega$ *be the level set, where* $\Omega \in \mathbb{R}^n$ *is a bounded closed set.*

Assumption 3.2. *There exists a positive constant m such that* $d^T B_k d \geq m\|d\|^2$, $\forall d \in \mathbb{R}^n$.

Remark 3.1. Combining with Assumptions 3.1 and 3.2, $f(x)$ is a quadratic continuous differentiable function. It follows that there exists a positive constant $M > m$, such that $\|B_k\| \leq M$.

Lemma 3.1. *Let the sequence* $\{x_n\}$ *be generated by the Algorithm 2.1. There are*

$$m_k(0) - m_k(d_k) \geq \frac{1}{2}\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}, \tag{3.1}$$

$$g_k^T d_k \leq -\frac{1}{2}\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\} \leq 0. \tag{3.2}$$

Lemma 3.2. *Let the sequence* $\{x_n\}$ *be generated by the Algorithm 2.1. Then,*

$$R_{k+1} \geq f_{k+1}$$

holds for all k.

Proof. (1) From the definition of R_{k+1} and p_k, if $r_k \geq \tau$, then we have the following:

$$R_{k+1} - f_{k+1} = (1 - p_k)(R_k - f_{k+1}) = 0.$$

(2) From $T_{k+1} = \beta T_k$, if $r_k < \tau$ and $p_k = exp\{-\frac{\tau - r_k}{T_k}\} \geq l_k$, then we can have $\lim_{k\to\infty} T_k = 0$. It is known that $l_k \in [e^{-v}, e^{-1/v}]$, and we can obtain $-v \leq lnl_k \leq -\frac{1}{v}$. Therefore, there exists a positive integer N such that $0 < -T_k lnl_k < \varepsilon_1$ holds for any $k > N$, $\varepsilon_1 > 0$. From the definition of p_k, we have the following:

$$r_k \geq \tau + T_k lnl_k \geq \tau - \varepsilon_1.$$

From the definition of r_k and (3.1), we have the following:

$$\begin{aligned} R_k - f_{k+1} &\geq (\tau - \varepsilon_1)[m_k(0) - m_k(d_k)] \\ &\geq \frac{1}{2}(\tau - \varepsilon_1)\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}. \end{aligned} \tag{3.3}$$

Let $\varepsilon_1 = \frac{\tau}{2}$; then

$$\begin{aligned} R_{k+1} - f_{k+1} &= (1 - p_k)(R_k - f_{k+1}) \\ &\geq \frac{1}{4}(1 - p_k)\tau\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\} \geq 0. \end{aligned}$$

(3) If $r_k < \tau$ and $p_k \leq l_k$, then $f\left(x_k + \lambda^{i_m} d_k\right) \leq R_k + \delta\lambda^{i_m} g_k^T d_k$ holds. Combining with (3.2), we obtain the following:

$$R_k - f_{k+1} \geq -\delta\lambda^{i_m} g_k^T d_k \geq 0.$$

Therefore, $R_{k+1} \geq f_{k+1}$ holds. The proof is completed. □

Lemma 3.3. *Let the sequence $\{x_n\}$ be generated by the Algorithm 2.1. The sequence $\{R_k\}$ is monotonically decreasing.*

Proof. (1) If $r_k \geq \tau$, namely,

$$R_k - f_{k+1} \geq \tau\left[m_k(0) - m_k(d_k)\right] \geq \frac{1}{2}\tau\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\},$$

then

$$f_{k+1} \leq R_k - \frac{1}{2}\tau\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}.$$

According to the definition of R_{k+1}, we have the following:

$$R_{k+1} \leq R_k - \frac{1}{2}\tau p_k\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\} < R_k. \tag{3.4}$$

(2) By (3.3) and the definition of R_{k+1}, if $r_k < \tau$ and $p_k > l_k$, then we obtain the following:

$$R_k - R_{k+1} = p_k\left(R_k - f_{k+1}\right) > \frac{1}{4}\tau p_k\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}. \tag{3.5}$$

(3) If $r_k < \tau$ and $p_k \leq l_k$, then $f\left(x_k + \lambda^{i_m} d_k\right) \leq R_k + \delta\lambda^{i_m} g_k^T d_k$ holds. Since $g_k^T d_k \leq 0$, then $f_{k+1} \leq R_k$. By the definition of R_{k+1}, we obtain the following:

$$R_{k+1} = (1 - p_k) R_k + p_k f_{k+1} \leq (1 - p_k) R_k + p_k R_k = R_k.$$

Therefore, $R_{k+1} \leq R_k$ holds. The proof is completed. □

Lemma 3.4. *Let the sequence $\{x_n\}$ be generated by the Algorithm 2.1. Then, the step length α_k satisfies $\alpha_k > \frac{(1-\delta)\lambda m}{M}$.*

Proof. The proof can be proved similarly to the proof of Lemma 3.3 in [13] and is omitted here. □

To facilitate the subsequent discussion, the definition of the sequence M_k is introduced as $M_k = 1 + \max\limits_{0 \leq i \leq k} \|B_i\|$.

Lemma 3.5. *Assume that the sequences $\{x_k\}$ and $\{R_k\}$ are generated by the Algorithm 2.1. If there exists a positive constant ε, such that $\|g_k\| \geq \varepsilon$ holds, then for any k, there is the following:*

$$R_{k+1} - R_k \leq -\frac{1}{2}\varepsilon p_k \psi \min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\},$$

where $\psi = \min\left\{\frac{1}{2}\tau, \frac{\delta(1-\delta)\lambda m}{M}\right\}$.

Proof. (1) By (3.4) and (3.5), if $p_k > l_k$, then we know that

$$R_{k+1} - R_k \leq -\frac{1}{4}\tau p_k \|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}.$$

(2) By combining (3.2), Lemma 3.4, and $f_{k+1} \leq R_k + \delta\alpha_k g_k^T d_k$, if $p_k \leq l_k$, then the following holds:

$$f_{k+1} \leq R_k - \frac{1}{2}\frac{(1-\delta)\lambda m}{M}\delta \|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}.$$

According to the definition of R_{k+1}, we obtain the following:

$$R_{k+1} - R_k = p_k (f_{k+1} - R_k) \leq -\frac{1}{2}\frac{(1-\delta)\lambda m}{M}\delta p_k \|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\}.$$

Additionally, if $\psi = \min\left\{\frac{1}{2}\tau, \frac{\delta(1-\delta)\lambda m}{M}\right\}$ and $\|g_k\| \geq \varepsilon$, then

$$R_{k+1} - R_k \leq -\frac{1}{2}\varepsilon p_k \psi \min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\}.$$

The proof is completed. □

Lemma 3.6. *Assume that Assumption 3.1 holds. If the sequence $\{x_k\}$ generated by the Algorithm 2.1 does not converge (i.e., there exists a positive constant ε, such that $\|g_k\| \geq \varepsilon$), then*

$$\lim_{k\to\infty} \min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\} = 0.$$

Proof. Algorithm 2.1 shows that $R_0 = f_0$. From Lemmas 3.2 and 3.3, we know that $f_{k+1} \leq R_{k+1} \leq R_k \leq f_0$. Assumption 3.1 shows that if $\{f(x_k)\}$ is bounded, then $\{R_k\}$ is bounded. According to Lemma 3.5, we can know that $\lim_{k\to\infty} \min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\} = 0$. □

Lemma 3.7. *If the sequence $\{x_k\}$ generated by the Algorithm 2.1 does not converge (i.e., there exists a positive constant ε such that $\|g_k\| \geq \varepsilon$), then there exists a set $J = \{k|p_k < l_k\}$ such that the following inequality holds for a sufficiently large $k \in J$:*

$$\|x_{k+1} - x_k\| \geq \sqrt{\frac{(1-\tau)\varepsilon \min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\}}{M - \tau m}}, \qquad \alpha_k = 1, \tag{3.6}$$

$$\|x_{k+1} - x_k\| > \frac{(1-\delta)\varepsilon\lambda}{M} \min\left\{1, \frac{\varepsilon}{\Delta_k M_k}\right\}, \qquad \alpha_k < 1, \tag{3.7}$$

$$\Delta_k > \frac{\varepsilon}{M_k}, \qquad \forall k \in J.$$

Proof. Based on α, we consider the following two cases.
Case 1. If $i_m = 0$ (i.e., $\alpha_k = 1$), then

$$\|x_{k+1} - x_k\| = \|d_k\| \leq \Delta_k, \forall k \in J.$$

At this point, there is $r_k < \tau$. According to $f_{k+1} \leq R_{k+1}$ and the definition of r_k, we have the following:

$$f(x_k) - f(x_k + d_k) \leq R_k - f(x_k + d_k) \leq \tau\left(-g_k^T d_k - \frac{1}{2} d_k^T B_k d_k\right). \tag{3.8}$$

According to the Taylor's expansion and Remark 3.1, we obtain the following:

$$f(x_k + d_k) - f(x_k) \leq g_k^T d_k + \frac{1}{2} M \|d_k\|^2. \tag{3.9}$$

Combining (3.8) and (3.9), we have the following:

$$-g_k^T d_k - \frac{1}{2} M \|d_k\|^2 \leq \tau\left(-g_k^T d_k - \frac{1}{2} d_k^T B_k d_k\right). \tag{3.10}$$

Combining (3.10) and Assumption 3.2, we obtain the following:

$$-(1-\tau) g_k^T d_k \leq \frac{1}{2}(M - \tau m)\|d_k\|^2. \tag{3.11}$$

Combining (3.2) and (3.11), we have the following:

$$(1-\tau)\|g_k\| \min\left\{\Delta_k, \frac{\|g_k\|}{\|B_k\|}\right\} \leq (M - \tau m)\|d_k\|^2. \tag{3.12}$$

From the definition of M_k and $\|g_k\| \geq \varepsilon$, and combined with $\|x_{k+1} - x_k\| = \|d_k\|$ and (3.12), we obtain the following:

$$(1-\tau)\varepsilon \min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\} \leq (M - \tau m)\|x_{k+1} - x_k\|^2.$$

Therefore, it is proved that $\|x_{k+1} - x_k\| \geq \sqrt{\frac{(1-\tau)\varepsilon \min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\}}{M - \tau m}}, \alpha_k = 1$.

Assuming $\Delta_k \leq \frac{\varepsilon}{M_k}$, there is (3.6) to obtain $\Delta_k \geq \sqrt{\frac{(1-\tau)\varepsilon\Delta_k}{M-\tau m}}$ (i.e., $\Delta_k \geq \frac{(1-\tau)\varepsilon}{M-\tau m}$). Combined with $\Delta_k \leq \frac{\varepsilon}{M_k}$, there is $\frac{\varepsilon}{M_k} \geq \Delta_k \geq \frac{(1-\tau)\varepsilon}{M-\tau m}$. This contradicts Lemma 3.6, so we have $\Delta_k > \frac{\varepsilon}{M_k}$.
Case 2. If $i_m > 0$ (i.e., $\alpha_k < 1$), then by combining (3.2), Remark 3.1, Lemma 3.2, and the Step 4 of Algortihm 2.1, according to the Taylor's expansion yield,

$$\begin{aligned} 0 &> R_k - f\left(x_k + \lambda^{-1}\alpha_k d_k\right) + \delta\lambda^{-1}\alpha_k g_k^T d_k \\ &\geq f_k - f\left(x_k + \lambda^{-1}\alpha_k d_k\right) + \delta\lambda^{-1}\alpha_k g_k^T d_k \\ &\geq -(1-\delta)\lambda^{-1}\alpha_k g_k^T d_k - \frac{1}{2} M \lambda^{-2} {\alpha_k}^2 \|d_k\|^2 \end{aligned}$$

$$\begin{aligned} &\geq \frac{1}{2}\lambda^{-1}\alpha_k\left[(1-\delta)\,\varepsilon\min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\} - M\lambda^{-1}\|d_k\|\|x_{k+1}-x_k\|\right] \\ &\geq \frac{1}{2}\lambda^{-1}\alpha_k\left[(1-\delta)\,\varepsilon\min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\} - M\lambda^{-1}\Delta_k\|x_{k+1}-x_k\|\right] \\ &= \frac{1}{2}\lambda^{-1}\alpha_k\Delta_k\left[(1-\delta)\,\varepsilon\min\left\{1, \frac{\varepsilon}{\Delta_k M_k}\right\} - M\lambda^{-1}\|x_{k+1}-x_k\|\right]. \end{aligned}$$

Therefore, it is proved that $\|x_{k+1}-x_k\| > \frac{(1-\delta)\varepsilon\lambda}{M}\min\left\{1, \frac{\varepsilon}{\Delta_k M_k}\right\}, \alpha_k < 1$.

There is

$$\|x_{k+1}-x_k\| = \alpha_k\|d_k\| \leq \Delta_k. \tag{3.13}$$

Suppose there exists $\Delta_k \leq \frac{\varepsilon}{M_k}$. Combining (3.7), we have the following:

$$\|x_{k+1}-x_k\| > \frac{(1-\delta)\,\varepsilon\lambda}{M}. \tag{3.14}$$

Combining (3.13), (3.14), and Assumption 3.1, there are $\frac{\varepsilon}{M_k} \geq \Delta_k \geq \|x_{k+1}-x_k\| > \frac{(1-\delta)\varepsilon\lambda}{M}$. This contradicts $\lim\limits_{k\to\infty}\min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\} = 0$. Therefore, $\Delta_k > \frac{\varepsilon}{M_k}$ holds.

In summary, the proof is completed. □

Lemma 3.8. *If the sequence $\{x_k\}$ generated by the Algorithm 2.1 does not converge (i.e., there exists a positive constant ε such that $\|g_k\| \geq \varepsilon$), then $\Delta_k > \frac{\varepsilon}{M_k}$ holds for all sufficiently large k.*

Proof. (1) If there are only finitely many k such that $\Delta_{k+1} \leq \Delta_k$ (i.e., J is a finite set), then there exists a positive constant Δ^* such that $\Delta_k > \Delta^*$. According to Lemma 3.6, we know that $\lim\limits_{k\to\infty}\frac{\varepsilon}{M_k} = 0$.

(2) We assume that J is an infinite set. It follows from Lemma 3.7 that there exists $\bar{k} \in J$, such that $\Delta_k > \frac{\varepsilon}{M_k}$ holds when $\forall k \in J, k \geq \bar{k}$. When $k \notin J$, $k \geq \bar{k}$, noted as $\tilde{k} = \max\{i | i \in J, i \leq k\}$. We have $\Delta_{\tilde{k}} > \frac{\varepsilon}{M_{\tilde{k}}}$, and $\tilde{k}+s \notin J$, $s = 1, 2, \cdots, k-\tilde{k}$. We can obtain $\Delta_{\tilde{k}} \leq \Delta_{\tilde{k}+1} \leq \Delta_{\tilde{k}+2} \leq \cdots \leq \Delta_k$, so $\Delta_k > \frac{\varepsilon}{M_{\tilde{k}}}$ holds. According to the definition of M_k, $\{M_k\}$ is a monotonically increasing sequence.

The proof is completed. □

Theorem 1. *If Assumptions 3.1 and 3.2 hold, and $\{B_k\}$ satisfies $\sum\limits_{k=0}^{\infty}\frac{1}{M_k} = \infty$, then the sequence $\{x_k\}$ generated by the Algorithm 2.1 satisfies the following:*

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

Proof. Suppose there exists a positive constant ε such that $\|g_k\| \geq \varepsilon$. Lemma 3.8 holds, that is $\Delta_k > \frac{\varepsilon}{M_k}$ holds for sufficiently large k. By Lemma 3.5, we know that $R_k - R_{k+1} \geq \frac{1}{2}\varepsilon p_k\psi\min\left\{\Delta_k, \frac{\varepsilon}{M_k}\right\}$. We have $\sum\limits_{k=0}^{\infty}(R_k - R_{k+1}) \geq \sum\limits_{k=0}^{\infty}\frac{1}{2}\varepsilon^2 p_k\psi\frac{1}{M_k}$, then $\sum\limits_{k=0}^{\infty}\frac{1}{M_k} < \infty$ contradicts $\sum\limits_{k=0}^{\infty}\frac{1}{M_k} = \infty$. The theorem is proved. □

4. Numerical experiments

In the following, we will show the effectiveness of our Algorithm 2.1 through some numerical experimental results. The numerical experiments are all implemented by using MATLAB (R2017a) on a PC with a CPU of 2.30 GHz and 8.00 GB RAM. The relevant parameters in Algorithm 2.1 and other comparison algorithms are selected as $T_0 = 200$, $\nu = 100$, $\varepsilon = 10^{-5}$, $\Delta_0 = \|g_0\|$, $\lambda = 0.5$, $\delta = 0.5$, $\eta_1 = 0.25$, $\eta_2 = 0.75$,$\gamma_1 = 0.5$, $\gamma_2 = 2$, $\beta = 0.9$, and $\tau = 0.25$. The matrix B_k is updated by the BFGS formula, i.e.,

$$B_{k+1} = B_k + \frac{y_k y_k^T}{s_k^T y_k} - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k}.$$

In order to explore the influence of the simulated annealing idea, nonmonotone technologies, and different trust region ratios on the algorithm, we perform the following three sets of comparison experiments:

(a) Algorithm 2.1 is compared with the existing trust region algorithm. The NTRM (Nonmonotone Trust Region Method With Line Search) algorithm [13] is a classical nonmonotone trust region algorithm. The SATRBB (Simulated Annealing-based Trust Region Bazilai-Borwein) algorithm [20] is a trust region algorithm combined with simulated annealing.

(b) Algorithm 2.1 is compared with the algorithms that combine different line search techniques (i.e., Algorithm 2.1 without a line search (SATR) and with a monotonic line search (ATR)).

(c) Algorithm 2.1 is compared with algorithms that combine different ratios (i.e., Algorithm 2.1 by using the original trust region ratio (SATR-OR), correcting the numerator portion of the trust region ratio to the form in [13] (SATR-U), and correcting the numerator and denominator portions of the trust region ratio according to C_k in [13] (SATR-UD)).

Remark 4.1. The SATR-UD algorithm is inspired by Remark in [29], and the form of trust region ratio is as follows:

$$r_k = \frac{C_k - f(x_k + d_k)}{C_k - f_k - m_k(d_k)}.$$

The test functions used and their associated dimensionality are shown in Table 1. We require the maximum number of iterations of the algorithm to be 5000. It is worth noting that the algorithm which uses the idea of simulated annealing is random in practice; therefore the relevant data are the average of the results of 10 tests.

To further investigate the efficiency of the algorithm, we use the performance profiles [9] to evaluate and compare the performance of the solvers. Assume that k denotes the number of iterations required, and t denotes the time required for the algorithm. Moreover, there exist n_s solvers on the test set S and n_p problems on test set P. If the computation running time is used as the performance metric, then we define $t_{p,s}$ as computing time required to solve problem p by the solver s. We define the performance ratio as follows:

$$r_{p,s_0} = \frac{t_{p,s_0}}{\min\left\{t_{p,s} : s \in S\right\}}.$$

There exists a r_M such that there is $r_M \geq r_s$ for any p and s. For the overall evaluation of the solver, we define the following:

$$\rho_s(\tau) = \frac{1}{n_p} size\left\{p \in P : r_{p,s} \leq \tau\right\},$$

where ρ is the cumulative distribution function of the performance ratio, which is the probability that the performance ratio $r_{p,s}$ of the solver s is within the best possible ratio factor τ.

Table 1. List of test functions.

Problem	*Dimension*	*Problem*	*Dimension*
Helical valley	3	*Chebyquad*	7
*Biggs EXP*6	6	*Freudenstein and Roth*	2
Gaussian	3	*Generalized Rosebrock*	50/100/500
Powell badly scaled	2	*Boundary value*	100/500
Box	3	*Broyden tridiagonal*	1000/2000
Variable dimension	1000/3000/5000	*Separable cubic*	100/500/1000
Watson	1000/2000	*Nearly separable*	1000/2000/3000/5000
*Penalty*1	500/1000/1500	*Yang tridiagonal*	50/100/500
Brown and Dennis	4	*Allgower*	100/300/500
Wood	4	*Schittkowski*	100/300
Extended Rosenbrock	500/1000	*Beale*	2
Extended Powell singular	100/500/1000		

The above three sets of experiments were conducted on the efficiency of solving some problems using the number of iterations and the time required for the algorithm as the performance metrics. The results are shown in Figures 1–3. It is not difficult to see from Figures 1 and 2 that Algorithm 2.1 has good properties compared with existing algorithms, and the nonmonotone line search technology proposed in this paper greatly improves the performance of the algorithm. Figure 3 shows that although the SATR-UD algorithm has a slight advantage for some problems. However, compared with Algorithm 2.1, there are still some problems that the SATR-UD algorithm cannot solve. All in all, the algorithm proposed in this paper is the preferred solver for the above problem with a high probability. Algorithm 2.1 solves the problems with a higher accuracy and requires fewer iterations.

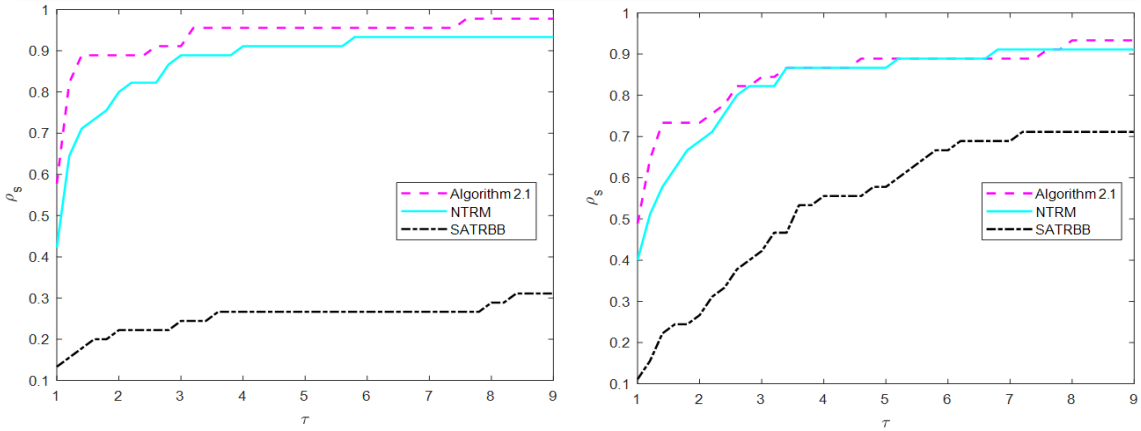

Figure 1. Performance profiles of (a) for the number of iterations k and the CPU time t.

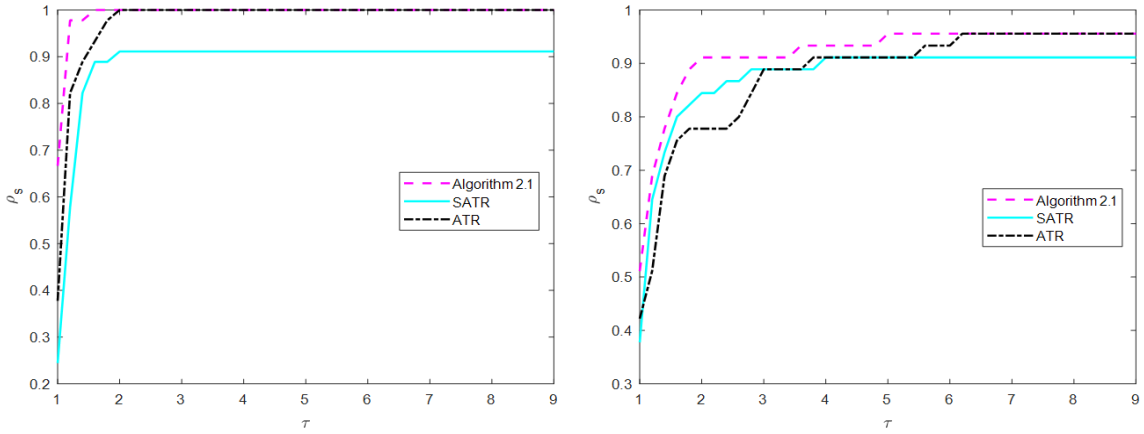

Figure 2. Performance profiles of (b) for the number of iterations k and the CPU time t.

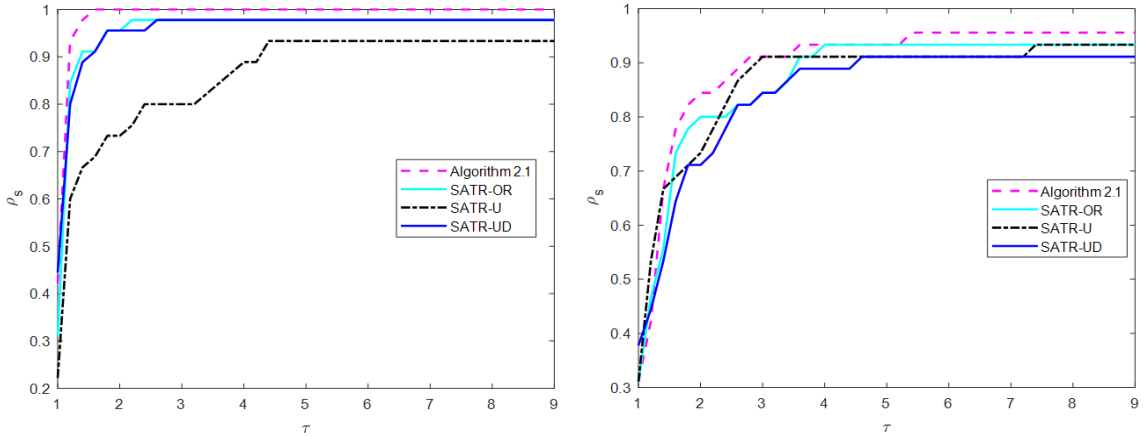

Figure 3. Performance profiles of (c) for the number of iterations k and the CPU time t.

5. Conclusions

In this paper, we proposed a novel stochastic nonmonotone trust region algorithm, which incorporated the simulated annealing for an enhanced performance. This integration of ideas aimed to address some of the challenges faced by traditional optimization methods, especially in terms of avoiding the local minima and achieving a global convergence. At the heart of our algorithm lies the construction of a new sequence, which was governed by the Metropolis criterion. The trust region ratio was modified based on this sequence, and the iterative criterion was adjusted according to the improved ratio. When the current trial step was not accepted, the iteration step size was obtained based on the modified nonmonotone line search. By doing so, we not only reduced the number of solutions required for the trust region subproblem, but also enhanced the algorithm's ability to escape the local minimum and converge to a global optimum. A theoretical analysis showed that our proposed algorithm achieved a global convergence under certain conditions, making it avoid local optimal solutions as much as possible. Furthermore, numerical experiments were conducted to demonstrate the effectiveness of our algorithm in practical applications. These experiments revealed that our algorithm outperformed traditional methods.

Author contributions

Yiting Zhang: Conceptualization, Methodology, Investigation, Software, Validation, Writing-original draft; Chongyang He: Funding acquisition, Investigation, Software; Wanting Yuan: Validation, Writing-review & editing; Mingyuan Cao: Funding acquisition, Methodology, Software, Writing-review & editing. All authors have read and approved the final version of the manuscript for publication.

Acknowledgments

The key project of the natural science foundation joint fund of Jilin province (YDZJ202201ZYTS303, YDZJ202101ZYTS167, 20230508184RC, JJKH20200037KJ); The graduate innovation project of Beihua University (2023003); The science and technology development plan project of Jilin province (20200403193SF).

Conflict of interest

All authors declare no conflicts of interest in this paper.

References

1. N. M. Alexandrov, J. E. Dennis, R. M. Lewis, V. Torczon, A trust region framework for managing the use of approximation models in optimization, *Struct. Optim.*, **15** (1998), 16–23. https://doi.org/10.1007/BF01197433
2. S. Babaie-Kafaki, R. Ghanbari, N. Mahdavi-Amiri, An efficient and practically robust hybrid metaheuristic algorithm for solving fuzzy bus terminal location problems, *Asia-Pac. J. Oper. Res.*, **29** (2012), 1250009. https://doi.org/10.1142/S0217595912500091
3. S. Babaie-Kafaki, S. Rezaee, A randomized nonmonotone adaptive trust region method based on the simulated annealing strategy for unconstrained optimization, *Int. J. Intell. Comput.*, **12** (2019), 389–399. https://doi.org/10.1108/IJICC-12-2018-0178
4. J. Bai, L. Jia, Z. Peng, A new insight on augmented Lagrangian method with applications in machine learning, *J. Sci. Comput.*, **99** (2024), 53. https://doi.org/10.1007/s10915-024-02518-0
5. J. Chen, W. Y. Sun, Nonmonotone adaptive trust-region algorithms with indefinite dogleg path for unconstrained minimization, *Northeast. Math. J.*, **24** (2008), 19–30.
6. J. Deepho, A. B. Abubakar, M. Malik, I. K. Argyros, Solving unconstrained optimization problems via hybrid CD-DY conjugate gradient methods with applications, *J. Comput. Appl. Math.*, **405** (2022), 113823. https://doi.org/10.1016/j.cam.2021.113823
7. N. Y. Deng, Y. Xiao, F. J. Zhou, Nonmonotonic trust region algorithm, *J. Optim. Theory Appl.*, **76** (1993), 259–285. https://doi.org/10.1007/BF00939608
8. S. Di, W. Sun, A trust region method for conic model to solve unconstraind optimizaions, *Optim. Method. Softw.*, **6** (1996), 237–263. https://doi.org/10.1080/10556789608805637

9. E. D. Dolan, J. J. More, Benchmarking optimization software with performance profiles, *Math. Program.*, **91** (2002), 201–213. https://doi.org/10.1007/s101070100263

10. J. Fu, W. Sun, Nonmonotone adaptive trust-region method for unconstrained optimization problems, *Appl. Math. Comput.*, **163** (2005), 489–504. https://doi.org/10.1016/j.amc.2004.02.011

11. E. M. Gertz, *Combination trust-region line search methods for unconstrained optimization*, San Diego: University of California, 1999.

12. L. Grippo, F. Lampariello, S. Lucidi, A nonmonotone line search technique for Newton's method, *SIAM J. Numer. Anal.*, **23** (1986), 707–716. https://doi.org/10.1137/0723046

13. N. Gu, J. Mo, Incorporating nonmonotone strategies into the trust region method for unconstrained optimization, *Comput. Math. Appl.*, **55** (2008), 2158–2172. https://doi.org/10.1016/j.camwa.2007.08.038

14. M. Hatamian, M. Paripour, F. M. Yaghoobi, N. Karamikabir. An adaptive nonmonotone line search technique for solving systems of nonlinear equations, *J. Math.*, **2021** (2021), 7134561. https://doi.org/10.1155/2021/7134561

15. D. Henderson, S. H. Jacobson, A. W. Johnson, The theory and practice of simulated annealing, In: *Handbook of Metaheuristics*, Boston: Springer, 2003, 287–319. https://doi.org/10.1007/0-306-48056-5_10

16. S. Huang, Z. Wan, J. Zhang, An extended nonmonotone line search technique for large-scale unconstrained optimization, *J. Comput. Appl. Math.*, **330** (2018), 586–604. https://doi.org/10.1016/j.cam.2017.09.026

17. S. Kirkpatrick, C. D. Gellat, M. P. Vecchi, Optimization by simulated annealing, *Science*, **220** (1983), 671–680. https://doi.org/10.1126/science.220.4598.671

18. J. Lee, K. Jung, J. H. Kwon, The aerodynamic shape optimization of airfoils using unconstrained trust region methods, *Eng. Optimiz.*, **41** (2009), 459–471. https://doi.org/10.1080/03052150802596068

19. T. Li, Z. Wan, J. Guo, A new nonmonotone spectral projected gradient algorithm for box-constrained optimization problems in m×n real matrix space with application in image clustering, *J. Comput. Appl. Math.*, **438** (2024), 115563. https://doi.org/10.1016/j.cam.2023.115563

20. X. Li, W. Dong, Z. Peng, A new nonmonotone trust region Barzilai-Borwein method for unconstrained optimization problems, *Acta Math. Appl. Sin. Engl. Ser.*, **37** (2021), 166–175. https://doi.org/10.1007/s10255-021-0997-9

21. Y. Liu, X. Liu, Application and performances of unconstrained optimization methods in seafloor AVO inversion, *Arab. J. Geosci.*, **9** (2016), 652. https://doi.org/10.1007/s12517-016-2692-3

22. S. Lu, Z. Wei, L. Li, A trust region algorithm with adaptive cubic regularization methods for nonsmooth convex minimization, *Comput. Optim. Appl.*, **51** (2012), 551–573. https://doi.org/10.1007/s10589-010-9363-1

23. Q. Ni, Optimality conditions for trust-region subproblems involving a conic model, *SIAM J. Optim.*, **15** (2005), 826–837. https://doi.org/10.1137/S1052623402418991

24. T. D. Niri, M. Heydari, M. M. Hosseini, Two nonmonotone trust region algorithms based on an improved Newton method, *J. Appl. Math. Comput.*, **64** (2020), 179–194. https://doi.org/10.1007/s12190-020-01350-7

25. J. Nocedal, Y. Yuan, Combining trust region and line search techniques, *Advances in nonlinear programming*, Boston, MA: Springer, 1998, 153–175. https://doi.org/10.1007/978-1-4613-3335-7_7

26. S. Rezaee, S. Babaie-Kafaki, An adaptive nonmonotone trust region algorithm, *Optim. method. softw.*, **34** (2019), 264–277. https://doi.org/10.1080/10556788.2017.1364738

27. S. Sun, J. Nocedal, A trust region method for noisy unconstrained optimization, *Math. Program.*, **202** (2023), 445–472. https://doi.org/10.1007/s10107-023-01941-9

28. W. Sun, J. Han, J. Sun, Global convergence of nonmonotone descent methods for unconstrained optimization problems, *J. Comput. Appl. Math.*, **146** (2002), 89–98. https://doi.org/10.1016/S0377-0427(02)00420-X

29. W. Sun, Nonmonotone trust region method for solving optimization problems, *Appl. Math. Comput.*, **156** (2004), 159–174. https://doi.org/10.1016/j.amc.2003.07.008

30. P. L. Toint, Non-monotone trust-region algorithms for nonlinear optimization subject to convex constraints, *Math. Program.*, **77** (1997), 69–94. https://doi.org/10.1007/BF02614518

31. Z. Wan, Y. Chen, S. Huang, D. Feng, A modified nonmonotone BFGS algorithm for solving smooth nonlinear equations, *Optim. Lett.*, **8** (2014), 1845–1860. https://doi.org/10.1007/s11590-013-0678-6

32. X. S. Yang, *Nature-inspired optimization algorithms*, Academic Press, 2020. https://doi.org/10.1016/C2013-0-01368-0

33. G. L. Yuan, Z. X. Wei, A trust region algorithm with conjugate gradient technique for optimization problems, *Numer. Func. Anal. Opt.*, **32** (2011), 212–232. https://doi.org/10.1080/01630563.2010.532273

34. H. Zhang, W. W. Hager, A nonmonotone line search technique and its application to unconstrained optimization, *SIAM J. Optim.*, **14** (2004), 1043–1056. https://doi.org/10.1137/S1052623403428208

35. Q. Y. Zhou, J. Chen, Z. W. Xie, A nonmonotone trust region method based on simple quadratic models, *J. Comput. Appl. Math.*, **272** (2014), 107–115. https://doi.org/10.1016/j.cam.2014.04.026

AIMS Press