J. Nonlinear Var. Anal. 11 (2027), No. 2, pp. 393-418
Available online at http://jnva.biemdas.com
https://doi.org/10.23952/jnva.11.2027.2.08

A TRUST-REGION ALGORITHM WITH A T_k-BASED NONMONOTONE STRATEGY FOR UNCONSTRAINED MULTIOBJECTIVE OPTIMIZATION

JIAN-WEN PENG[1], JIE REN[2], ABHISHEK SINGH[1,*], ELISABETH KÖBIS[3]

[1]School of Mathematical Sciences, Chongqing Normal University, Chongqing, China
[2]Chongqing Research Institute of Big Data, Peking University, Chongqing, China
[3]Department of Mathematical Sciences, Norwegian University of Science and Technology, Trondheim, Norway

Abstract. Trust-region methods provide a reliable framework for solving unconstrained multiobjective optimization problems, but classical monotone acceptance rules may lead to conservative steps and slow progress on challenging problems. This paper presents a trust-region algorithm based on a T_k-type nonmonotone strategy that introduces greater flexibility into the acceptance mechanism while preserving the convergence properties of the method. Under standard assumptions, we prove that every accumulation point generated by the algorithm satisfies the first-order Pareto criticality conditions. The proposed method is tested on a diverse set of benchmark problems and compared with three existing trust-region algorithms. The numerical results demonstrate competitive performance in terms of iterations, objective function evaluations, gradient evaluations, and CPU time, while maintaining high-quality Pareto approximations measured by Hypervolume (HV), Inverted Generational Distance Plus (IGD^+), and Purity. Performance profile analysis further confirms the efficiency and robustness of the proposed approach.

Keywords. Global convergence; Multiobjective optimization; Nonmonotone strategy; Pareto critical point; Trust-region methods.
2020 MSC. 90C53, 90C30, 49J53.

1. INTRODUCTION

We study the unconstrained multiobjective optimization problem $\min_{x\in\mathbb{R}^n} F(x)$, where $F(x) = \big(F_1(x), F_2(x), \ldots, F_m(x)\big)^\top$, and each component function $F_i : \mathbb{R}^n \to \mathbb{R}$ is twice continuously differentiable. Unlike scalar optimization, there is no single objective to minimize here. Instead, the objectives usually compete with each other, so the goal is to find Pareto optimal or Pareto critical points, which provide the natural notion of stationarity in vector optimization.

Multiobjective optimization arises in many practical problems, including engineering design, economics, scheduling, resource allocation, and decision making. Because the objectives are often conflicting, one cannot expect to improve all components simultaneously at every iteration. This has motivated the development of numerical methods that work directly with vector-valued objectives rather than reducing the problem to a scalar one. Several important methods have been proposed for this purpose. Fliege and Svaiter introduced a steepest descent framework for

*Corresponding author.
E-mail address: abhisheks.mat16@gmail.com (A. Singh).
Received 10 February 2026; Accepted 20 August 2026; Published online 20 November 2026.
©2027 Journal of Nonlinear and Variational Analysis

multicriteria optimization and showed how Pareto criticality can be used as the basic stationarity concept [13]. Later, Newton-type methods were developed for multiobjective optimization [14, 19, 23, 24], together with projected gradient methods [18], gradient-free methods [35], and nonlinear conjugate gradient methods [26, 28, 32]. These methods provide effective tools for solving vector optimization problems while preserving meaningful convergence properties.

For nonconvex problems, globalization becomes especially important. Trust-region methods are among the most reliable globalization strategies because they combine local model information with a safeguard on the step length. Their scalar theory is well established [7]. In the multiobjective setting, Arias et al. developed a trust-region globalization strategy for nonconvex unconstrained multiobjective optimization and proved global convergence results [5]. This work showed that trust-region ideas can be successfully adapted to the vector-valued case. At the same time, monotone decrease conditions can be too restrictive in practice. When a problem is highly nonlinear or nonconvex, requiring every accepted step to improve every objective can lead to very small steps and slow progress. This is why nonmonotone techniques have attracted so much attention. In scalar optimization, Zhang and Hager proposed a well-known nonmonotone line search strategy [36], and nonmonotone trust-region methods were later studied by Ahookhosh and Amini [3], Amini and Ahookhosh [1], Ahookhosh and Ghaderi [2]. These ideas allow some temporary increase in the objective values while still maintaining convergence.

Motivations and contributions. Nonmonotone ideas were also extended to multiobjective optimization. Mita et al. [27] developed nonmonotone line searches for unconstrained multiobjective problems and established their convergence behavior. Yang and Zhou [33] combined a trust-region framework with a nonmonotone technique for nonconvex multiobjective optimization. These results suggest that nonmonotone acceptance rules can improve flexibility and practical performance without sacrificing convergence.

Motivated by these developments, we propose a new trust-region algorithm for unconstrained nonconvex multiobjective optimization based on a T_k-type nonmonotone reference sequence. The idea is to keep the stability of the trust-region framework while allowing a controlled amount of nonmonotonicity in the acceptance test. Compared with purely monotone strategies, this gives the algorithm more freedom to move through difficult regions of the search space. At the same time, the method remains directly within the multiobjective framework and does not require scalarization or weight selection.

The main contributions of this paper are as follows:

- We introduce a trust-region algorithm for unconstrained nonconvex multiobjective optimization that incorporates a T_k-type nonmonotone reference sequence.
- We establish the well-definedness of the algorithm and prove its global convergence under standard assumptions.
- We present numerical experiments on benchmark multiobjective test problems and compare the proposed method with existing approaches.

The rest of this paper is organized as follows. Section 2 presents the notation, definitions, and assumptions used throughout the paper. Section 3 describes the proposed algorithm. Section 4 contains the convergence analysis. Section 5 reports numerical experiments. Section 6, the last section, concludes this paper.

2. PRELIMINARIES

Throughout this paper, we use the following notation:

- $\mathbb{R}^m_+$ denotes the set of all vectors in $\mathbb{R}^m$ whose components are nonnegative;
- $\mathbb{R}^m_{++}$ denotes the set of all vectors in $\mathbb{R}^m$ whose components are strictly positive;
- $\|\cdot\|$ stands for the Euclidean norm.

We consider the unconstrained multiobjective optimization problem

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

where $F(x) = \big[F_1(x), F_2(x), \ldots, F_m(x)\big]^\top : \mathbb{R}^n \to \mathbb{R}^m$. For each $i = 1,2,\ldots,m$, we assume that $F_i \in \mathbb{C}^2(\mathbb{R}^n)$ and F_i is bounded from below. We denote by $\nabla F_i(x) \in \mathbb{R}^n$ and $\nabla^2 F_i(x) \in \mathbb{R}^{n\times n}$ the gradient and Hessian of F_i at x, respectively. The Jacobian matrix of F at x is written as

$$JF(x) = \big(\nabla F_1(x), \nabla F_2(x), \ldots, \nabla F_m(x)\big)^\top.$$

Definition 2.1. [17] A point $x^* \in X$ is called a Pareto optimal, or efficient, solution of (2.1) if there is no $x \in \mathbb{R}^n$ such that $F(x) \le F(x^*)$ and $F(x) \ne F(x^*)$. Likewise, x^* is called a weakly Pareto optimal, or weakly efficient, solution if there is no $x \in \mathbb{R}^n$ satisfying $F(x) < F(x^*)$.

Definition 2.2. [13] A point $x^* \in \mathbb{R}^n$ is called a first-order Pareto critical point of problem (2.1) if there does not exist a direction $d \in \mathbb{R}^n$ such that $\nabla F_i(x^*)^\top d < 0$, $i = 1,2,\ldots,m$.

Theorem 2.1. [14] *Let $F \in C^2(X, \mathbb{R}^m)$. Then the following statements hold:*

(1) If $x^* \in X$ is weakly efficient, then x^* is a critical point.
(2) If $F_1, F_2, \ldots, F_m$ are convex on X and $x^* \in X$ is a critical point of F, then x^* is a weakly efficient solution.
(3) If $\nabla^2 F_i(x)$ is positive definite on X for each $i = 1,2,\ldots,m$, and $x^* \in X$ is a critical point, then x^* is an efficient solution.

3. TRUST-REGION ALGORITHM FOR NONMONOTONIC LINE SEARCH IN MULTIOBJECTIVE OPTIMIZATION

In this section, we combine a T_k-type nonmonotone strategy with a trust-region framework for unconstrained multiobjective optimization. The resulting method is designed to preserve the stabilizing effect of trust-region steps while allowing controlled nonmonotonicity in the line search.

We now consider the unconstrained multi-objective problem (2.1). Each objective function is locally approximated by its second-order Taylor model at the current iterate x_k:

$$F_i(x_k + d) \approx F_i(x_k) + \nabla F_i(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 F_i(x_k) d.$$

Accordingly, we define the quadratic model for the i^{th} objective at the current iterate x_k by $m_k^i(d) = \nabla F_i(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 F_i(x_k) d$. This model describes the local variation of the ith component and is used to measure the quality of the computed trial step. Therefore, to obtain a trial

step $d_k \in \mathbb{R}^n$ of F at x_k, we need to solve the following subproblem:

$$\begin{aligned} \min_{d\in\mathbb{R}^n} \quad & \max_{i=1,2,\ldots,m} \left(\nabla F_i(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 F_i(x_k) d \right) \\ \text{subject to} \quad & \|d\| \le \Delta_k. \end{aligned} \tag{3.1}$$

In this paper, we use $\phi(x_k)$ and d_k to represent the optimal value and the corresponding solution of problem (3.1) at point x_k, respectively. Thus,

$$\begin{aligned} \phi(x_k) &= \min_{\|d\|\le\Delta_k} \max_{i=1,2,\ldots,m} \left(\nabla F_i(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 F_i(x_k) d \right) \\ \text{and} \quad d_k &= \arg \min_{\|d\|\le\Delta_k} \max_{i=1,2,\ldots,m} \left(\nabla F_i(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 F_i(x_k) d \right). \end{aligned}$$

This model captures second-order information for each objective and forms the basis of the trust-region approach considered here. Since problem (3.1) is non-smooth, we introduce an auxiliary variable t to reformulate the subproblem (3.1) as follows:

$$\left.\begin{aligned} \min \quad & t \\ \text{subject to} \quad & \nabla F_i(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 F_i(x_k) d - t \le 0, \quad i = 1,2,\ldots,m, \\ & \nabla F_i(x_k)^\top d - t \le 0, \quad i = 1,2,\ldots,m, \\ & \|d\| \le \Delta_k, \\ & (t,d) \in \mathbb{R}\times\mathbb{R}^n. \end{aligned}\right\} \tag{3.2}$$

where $\Delta_k > 0$ is the trust-region radius at k^{th} iteration. The constraints $\nabla F_i(x)^\top d - t \le 0$, $i = 1,2,\ldots,m$ are introduced to promote common descent properties of the computed trial step and to facilitate the subsequent line-search procedure. In particular, whenever the optimal value of the subproblem is negative, the resulting trial step d_k is a common descent direction for all objective functions. Problem (3.2) can be viewed as a modified epigraph reformulation of the trust-region subproblem (3.1), augmented with additional linear constraints. These constraints are introduced to promote common descent properties of the computed trial step and facilitate the subsequent line-search procedure. By solving this subproblem, we compute a trial step $d_k \in \mathbb{R}^n$. The following Theorem 3.1 extends the corresponding property established in [5] to the modified trust-region subproblem (3.2).

Theorem 3.1. [5] *Let $\phi(x)$ denote the optimal value of subproblem (3.2), and d be an associated solution. Then:*

(1) $\phi(x) \le 0$ for all x,
(2) the following are equivalent: (a) x is not Pareto critical; (b) $\phi(x) < 0$; (c) $d \ne 0$,
(3) $\phi(x)$ is continuous and d is bounded on compact sets.

3.1. **Acceptance and rejection of trial step d_k.** In the monotone case, the Armijo condition is

$$F_i(x_k + \alpha_k d_k) \le F_i(x_k) + \sigma\alpha_k \nabla F_i(x_k)^\top d_k,$$

$i = 1,2,\ldots,m$, where $\sigma \in (0,1/2)$, and $\alpha_k > 0$ is the step length. For multiobjective optimization, the classical Armijo condition is replaced by the nonmonotone version

$$F_i(x_k + \alpha_k d_k) \le F_i^{\ell(k)} + \sigma\alpha_k \nabla F_i(x_k)^\top d_k,$$

$i = 1,2,\ldots,m$, where $k = 0,1,2,\ldots$ and $m(0) = 0$, and $m(k) \le \min\{m(k-1)+1, N\}$, where $N \ge 0$ is the nonmonotone memory length. For convenience, define $F_i^k := F_i(x_k)$. The nonmonotone reference value is then

$$F_i^{\ell(k)} = \max_{0 \le j \le m(k)} \{F_i^{k-j}\},$$

i.e., the largest objective value among the last $m(k)+1$ iterations. We now incorporate the T_k-type nonmonotone term. The corresponding acceptance condition is

$$F_i(x_k + \alpha_k d_k) \le T_i(x_k) + \sigma \alpha_k \nabla F_i(x_k)^\top d_k,$$

$i = 1,2,\ldots,m$, where $\sigma \in (0, 1/2)$. The reference value $T_i(x_k)$ is defined by

$$T_i(x_k) = \begin{cases} F_i^{l(k)}, & k < N, \\ \max\{\bar{T}_i(x_k), F_i(x_k)\}, & k \ge N, \end{cases}$$

where $\bar{T}_i(x_k)$ is initialized and updated as $\bar{T}_i(x_N) = F_i(x_N)$ and

$$\bar{T}_i(x_k) = (1-\eta_{k-1})F_i(x_k) + \eta_{k-1}\bar{T}_i(x_{k-1}) + \xi_k\big(F_i(x_{k-N}) - F_i(x_{k-N-1})\big), \quad k \ge N+1.$$

The sequence $\{\eta_k\}_{k\ge 0}$ is prescribed a priori and satisfies $0 \le \eta_k \le \eta_{\max} < 1$, $k = 0,1,2,\ldots$. Consequently, the coefficients $\xi_k = \prod_{j=k-N-1}^{k-1} \eta_j$, $k \ge N+1$, are well-defined. With this initialization, the recursion is well-defined for all iterations. The term $\bar{T}_i(x_k)$ reuses information from previous iterates and permits a controlled amount of nonmonotonicity without losing the descent structure needed in the analysis. For a trial step d_k, the agreement between the actual decrease and the predicted decrease is measured, for each $i = 1,2,\ldots,m$, by

$$r_i^k = \frac{T_i(x_k) - F_i(x_k + d_k)}{m_i^k(0) - m_i^k(d_k)}.$$

Here, the actual decrease is defined by

$$\mathrm{ared}_i(d_k) = T_i(x_k) - F_i(x_k + d_k),$$

and the predicted decrease is defined by $\mathrm{pred}_i(d_k) = m_i^k(0) - m_i^k(d_k)$, $i = 1,2,\ldots,m$. If $r_i^k \ge \mu_1$, $i = 1,2,\ldots,m$, then the trial step d_k is accepted. We set $x_{k+1} = x_k + d_k$. Otherwise, we perform a backtracking line search along d_k and set $x_{k+1} = x_k + \alpha_k d_k$, where $\alpha_k = \min\{\alpha_{k,1}, \alpha_{k,2}, \ldots, \alpha_{k,m}\}$, and $\alpha_{k,i}$ is step length computed in k^{th} iteration for i^{th} objective function. The step-length $\alpha_{k,i}$ is chosen as the largest element of $\alpha_{k,i} = \{s, \rho s, \rho^2 s, \ldots\}$ such that it satisfies

$$F_i(x_k + \alpha_{k,i} d_k) \le T_i(x_k) + \sigma \alpha_{k,i} \nabla F_i(x_k)^\top d_k,$$

$i = 1,2,\ldots,m$, where $s > 0$, $\rho \in (0,1)$, and $\sigma \in (0, 1/2)$.

Based on the above quantities, we propose the following algorithm.

Algorithm 1 Trust-region algorithm with nonmonotone line search for multiobjective optimization

Step 1 (Initialization step). Choose an initial point $x_0 \in \mathbb{R}^n$, an initial trust-region radius $\Delta_0 > 0$ with $\Delta_{\max} > \Delta_0$, and constants $0 < \mu_1 < \mu_2 < 1$, $0 < \gamma_1 < 1 < \gamma_2$, $0 < \sigma < \frac{1}{2}$, $\zeta \in (0,1)$, $\varepsilon > 0$. For step-length selection, we chose $s = 1$, $0 < \rho < 1$ and then it becomes $s\rho^j = \zeta^j(say)$. Choose a nonnegative integer N and a sequence $\{\eta_k\}_{k\geq 0}$ satisfying $0 \leq \eta_k \leq \eta_{\max} < 1$, $k = 0,1,2,\ldots$. Set the iteration counter $k = 0$.

Step 2 (Termination criterion). Compute $\nabla F_i(x_k)$ and $\nabla^2 F_i(x_k)$ for each $i = 1,2,\ldots,m$. Solve the trust-region subproblem (3.2) with $x = x_k$ and $\Delta = \Delta_k$ to obtain d_k. If $|\phi(x_k)| < \varepsilon$, then stop, and give the output x_k as an ε-precision approximate Pareto critical point. If the termination criterion is not met, go to **Step 3**.

Step 3 (Main iteration).

Step 2.1 (Quadratic models and trust-region ratios).
For each objective function F_i, compute $m_k^i(d) = \nabla F_i(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 F_i(x_k) d$ and evaluate $r_k^i = \frac{T_i(x_k) - F_i(x_k + d_k)}{m_k^i(0) - m_k^i(d_k)}$, $\quad i = 1,2,\ldots,m$.

Step 2.2 (Acceptance of the trial step).

- If $r_k^i \geq \mu_1$, for all i, $i = 1,2,\ldots,m$, then accept the trial step and define $x_{k+1} = x_k + d_k$. Go to Step 2.4.
- If $r_k^i < \mu_1$ for at least one i, $i = 1,2,\ldots,m$, we proceed to Step 2.3.

Step 2.3 (Nonmonotone line search).
Compute $\alpha_k = \min\{\alpha_{k,1}, \alpha_{k,2}, \ldots, \alpha_{k,m}\}$, where $\alpha_{k,i}$ is computed by

$$\alpha_{k,i} = \max\left\{\zeta^j : F_i(x_k + \zeta^j d_k) \leq T_i(x_k) + \sigma\zeta^j \nabla F_i(x_k)^\top d_k, \quad j = 0,1,\ldots\right\},$$

for each $i = 1,2,\ldots,m$. Then, x_{k+1} is computed by $x_{k+1} = x_k + \alpha_k d_k$. Now go to next Step 2.5.

Step 2.4 (Update of the trust-region radius).
The trust-region radius will be updated according to

$$\Delta_{k+1} = \begin{cases} \min\{\gamma_2\Delta_k, \Delta_{\max}\}, & \mu_2 \leq r_k^i, \quad \forall i, \\ \Delta_k, & \mu_1 \leq r_k^i, \quad \forall i, \quad \exists l \text{ such that } r_k^l < \mu_2, \\ \gamma_1\Delta_k, & \exists l \text{ such that } r_k^l < \mu_1. \end{cases}$$

Step 2.5 (Update iteration counter). Set $k = k+1$ and return to **Step 2**.

An iteration is called *very successful* if $r_k^i \geq \mu_2$, for all $i = 1,\ldots,m$. If $r_k^i \geq \mu_1$, for all $i = 1,\ldots,m$, and there exists at least one index l such that $r_k^l < \mu_2$, then the iteration is called *successful*. Otherwise, it is called *unsuccessful*.

4. Convergence Analysis

To facilitate the convergence analysis of Algorithm 1, we introduce $e_{i,k} := \xi_k\big(F_i(x_{k-N}) - F_i(x_{k-N-1})\big)$, $k \geq N+1$, $i = 1,\ldots,m$. Then the update of the reference term can be written as

$$\bar{T}_i(x_k) = (1 - \eta_{k-1})F_i(x_k) + \eta_{k-1}\bar{T}_i(x_{k-1}) + e_{i,k}.$$

To keep the proof complete while retaining the extra correction term, we make the following assumptions.

Assumption 4.1. The extended level set $L_E(x_0) = \{x \in \mathbb{R}^n : F_i(x) \le F_i(x_0) + E_i,\ i = 1,\dots,m\}$ is bounded.

Assumption 4.2. There exist constants $L_g > 0$ and $L_H > 0$ such that, for all $x \in L_E(x_0)$, $i = 1,\dots,m$, $\|\nabla F_i(x)\| \le L_g$ and $\|\nabla^2 F_i(x)\| \le L_H$.

Assumption 4.3. There exists a constant $\alpha_{\min} \in (0,1]$ such that the step length generated by the nonmonotone backtracking line search satisfies $\alpha_k \ge \alpha_{\min}$, $k = 0,1,2,\dots$.

Assumption 4.4. The sequence $\{\eta_k\}_{k\ge 0}$ satisfies $0 \le \eta_k \le \eta_{\max} < 1$, $k = 0,1,2,\dots$.

Assumption 4.5. The correction sequence is absolutely summable, i.e., $\sum_{k=N+1}^{\infty} |e_{i,k}| < \infty$, $i = 1,\dots,m$.

For convenience, let $E_i := \sum_{k=N+1}^{\infty} |e_{i,k}|$, $i = 1,\dots,m$.

Remark 4.1. Lemma 4.2 guarantees that the backtracking line search always produces an acceptable step length. Assumption 4.3 additionally requires these step lengths to be uniformly bounded away from zero, which is only used in proving Lemma 4.8.

We begin with the basic property of the trust-region subproblem.

Lemma 4.1. *Let $(\phi(x_k), d_k)$ be a solution of the subproblem (7). If Algorithm 1 has not terminated at iteration k, then $\phi(x_k) < 0$, $d_k \ne 0$, and $\nabla F_i(x_k)^\top d_k \le \phi(x_k) < 0$, $i = 1,\dots,m$. In particular, d_k is a common descent direction for all objective functions.*

Proof. If Algorithm 1 does not terminate at iteration k, then x_k is not Pareto critical. By Theorem 3.1, this implies $\phi(x_k) < 0$ and $d_k \ne 0$. Since $(\phi(x_k), d_k)$ is feasible for (7), $\nabla F_i(x_k)^\top d_k - t \le 0$, $i = 1,\dots,m$, imply $\nabla F_i(x_k)^\top d_k \le \phi(x_k) < 0$, $i = 1,\dots,m$. Hence, d_k is a common descent direction for all objective functions. □

The next result shows that the nonmonotone line search is well-defined.

Lemma 4.2. *If Algorithm 1 does not terminate at iteration k, then*

$$\mathscr{A}_k := \left\{ \zeta^j :\ F_i(x_k + \zeta^j d_k) \le T_i(x_k) + \sigma \zeta^j \nabla F_i(x_k)^\top d_k,\ i = 1,\dots,m,\ j = 0,1,2,\dots \right\}$$

is nonempty. Consequently, the backtracking line search in Step 2.3 is well-defined and terminates after finitely many reductions.

Proof. From Lemma 4.1, one has $\nabla F_i(x_k)^\top d_k < 0$, $i = 1,\dots,m$. Since each objective function F_i is continuously differentiable,

$$\lim_{\alpha \downarrow 0} \frac{F_i(x_k + \alpha d_k) - F_i(x_k)}{\alpha} = \nabla F_i(x_k)^\top d_k.$$

Because $0 < \sigma < 1$, it follows that, for every $i = 1,\dots,m$, there exists $\delta_i > 0$ such that $F_i(x_k + \alpha d_k) \le F_i(x_k) + \sigma\alpha \nabla F_i(x_k)^\top d_k$, $0 < \alpha \le \delta_i$. Since $T_i(x_k) \ge F_i(x_k)$, we obtain

$$F_i(x_k + \alpha d_k) \le T_i(x_k) + \sigma\alpha \nabla F_i(x_k)^\top d_k, \qquad 0 < \alpha \le \delta_i.$$

Let $\delta = \min_{1\le i\le m}\delta_i > 0$. Then the above inequality holds simultaneously for all $i = 1,\dots,m$ whenever $0 < \alpha \le \delta$. Since $\zeta \in (0,1)$, the sequence $\{\zeta^j\}_{j\ge 0}$ satisfies $\zeta^j \to 0$ as $j \to \infty$. Hence there exists an integer $j_0 \ge 0$ such that $\zeta^{j_0} \le \delta$. Thus $\zeta^{j_0} \in \mathscr{A}_k$. Hence $\mathscr{A}_k$ is nonempty. Since $\{\zeta^j\}_{j\ge 0}$ is a strictly decreasing sequence, there exists a smallest admissible index $j_* = \min\{j \ge 0 : \zeta^j \in \mathscr{A}_k\}$. It follows that ζ^{j_*} is the largest admissible step length. Consequently, the backtracking line search in Step 2.3 is well-defined and terminates after finitely many reductions. □

We next control the reference sequence and the iterations.

Lemma 4.3. *Let $(\phi(x_k), d_k)$ be the solution of subproblem (3.2). If Algorithm 1 does not terminate at iteration k, then, for every $i = 1,\dots,m$, $m_i^k(0) - m_i^k(d_k) \ge |\phi(x_k)| > 0$. Then the ratio*

$$r_i^k = \frac{T_i(x_k) - F_i(x_k + d_k)}{m_i^k(0) - m_i^k(d_k)}$$

is well-defined.

Proof. Since d_k is an optimal solution of (3.1), one has

$$\phi(x_k) = \min_{\|d\|\le\Delta_k} \max_{1\le i\le m} m_i^k(d) = \max_{1\le i\le m} m_i^k(d_k),$$

where $m_i^k(d) = \nabla F_i(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 F_i(x_k) d$. Hence, $m_i^k(d_k) \le \max_{1\le j\le m} m_j^k(d_k) = \phi(x_k)$, $i = 1,\dots,m$. Lemma 4.1 gives $\phi(x_k) < 0$. Moreover, $m_i^k(0) = \nabla F_i(x_k)^\top 0 + \frac{1}{2} 0^\top \nabla^2 F_i(x_k) 0 = 0$. Therefore,

$$m_i^k(0) - m_i^k(d_k) = -m_i^k(d_k) \ge -\phi(x_k) = |\phi(x_k)| > 0,$$

for every $i = 1,\dots,m$. Hence the denominator in the definition of r_i^k is strictly positive, which proves that r_i^k is well-defined. □

Lemma 4.4. *Let $\{x_k\}$ be generated by Algorithm 1. Then, for every $k \ge 0$ and every $i = 1,\dots,m$, $F_i(x_{k+1}) < T_i(x_k)$.*

Proof. The proof is split into two cases.

Case 1. The trial step is accepted.

In this case, $r_i^k \ge \mu_1$, $i = 1,\dots,m$. By the definition of r_i^k, we have

$$r_i^k = \frac{T_i(x_k) - F_i(x_k + d_k)}{m_i^k(0) - m_i^k(d_k)}.$$

Since $\mu_1 > 0$, we obtain $\frac{T_i(x_k) - F_i(x_k+d_k)}{m_i^k(0) - m_i^k(d_k)} \ge \mu_1$. It follows from Lemma 4.3 that $m_i^k(0) - m_i^k(d_k) > 0$. Multiplying both sides of the above inequality by the positive quantity $m_i^k(0) - m_i^k(d_k)$ yields

$$T_i(x_k) - F_i(x_k + d_k) \ge \mu_1 \big(m_i^k(0) - m_i^k(d_k)\big) > 0.$$

Since $x_{k+1} = x_k + d_k$, it follows that $F_i(x_{k+1}) < T_i(x_k)$.

Case 2. The trial step is rejected.

In this case, $x_{k+1} = x_k + \alpha_k d_k$, where the step length α_k satisfies $F_i(x_k + \alpha_k d_k) \le T_i(x_k) + \sigma\alpha_k \nabla F_i(x_k)^\top d_k$. It follows from Lemma 4.1 that $\nabla F_i(x_k)^\top d_k < 0$. Since $\sigma > 0$ and $\alpha_k > 0$, we have $\sigma\alpha_k \nabla F_i(x_k)^\top d_k < 0$. Therefore, $F_i(x_{k+1}) = F_i(x_k + \alpha_k d_k) < T_i(x_k)$. Since both cases hold, we conclude that $F_i(x_{k+1}) < T_i(x_k)$, $i = 1,\dots,m$. This completes the proof. □

Lemma 4.5. *Let $\{x_k\}$ be generated by Algorithm 1. Then, for every $i = 1,\dots,m$, $F_i(x_k) \le F_i(x_0)$, $k = 0,1,\dots,N$.*

Proof. We prove the result by induction on k.

For $k = 0$, the conclusion is obvious since $F_i(x_0) = F_i(x_0)$. Suppose that $F_i(x_j) \le F_i(x_0)$, $j = 0,1,\dots,k$, for some $k < N$. Since $T_i(x_k) = F_i^{\ell(k)} = \max_{0\le r\le m(k)} F_i(x_{k-r})$, and every objective value appearing in the above maximum satisfies $F_i(x_{k-r}) \le F_i(x_0)$, it follows that $T_i(x_k) \le F_i(x_0)$. It follows from Lemma 4.4 that $F_i(x_{k+1}) < T_i(x_k) = F_i(x_0)$. Hence, $F_i(x_{k+1}) \le F_i(x_0)$ and then $F_i(x_k) \le F_i(x_0)$, $k = 0,1,\dots,N$. This completes the proof. □

Lemma 4.6. *Let $\{x_k\}$ be generated by Algorithm 1. Then, for every $i = 1,\dots,m$, $F_i(x_k) \le T_i(x_k) \le F_i(x_0) + E_i$, $k = 0,1,2,\dots$.*

Proof. $F_i(x_k) \le T_i(x_k)$ immediately follows from the definition of $T_i(x_k)$. Next, we define $S_k := \sum_{r=N+1}^{k} |e_{i,r}|$, $k \ge N$, where $S_N := 0$. We prove by induction that, $k \ge N$,

$$F_i(x_k) \le F_i(x_0) + S_k,\ \bar{T}_i(x_k) \le F_i(x_0) + S_k,\ T_i(x_k) \le F_i(x_0) + S_k.$$

It follows from Lemma 4.5 that $F_i(x_N) \le F_i(x_0)$. Since $\bar{T}_i(x_N) = F_i(x_N)$, we obtain $\bar{T}_i(x_N) \le F_i(x_0) = F_i(x_0) + S_N$. Moreover,

$$T_i(x_N) = \max\{\bar{T}_i(x_N), F_i(x_N)\} = F_i(x_N) \le F_i(x_0) + S_N.$$

Hence the induction hypothesis holds for $k = N$. Now, let us suppose that

$$F_i(x_k) \le F_i(x_0) + S_k,\ \bar{T}_i(x_k) \le F_i(x_0) + S_k,\ T_i(x_k) \le F_i(x_0) + S_k.$$

It follows from Lemma 4.4 that $F_i(x_{k+1}) < T_i(x_k) \le F_i(x_0) + S_k \le F_i(x_0) + S_{k+1}$. In view of $\bar{T}_i(x_{k+1}) = (1-\eta_k)F_i(x_{k+1}) + \eta_k \bar{T}_i(x_k) + e_{i,k+1}$, we obtain

$$\begin{aligned}\bar{T}_i(x_{k+1}) &\le (1-\eta_k)(F_i(x_0) + S_k) + \eta_k(F_i(x_0) + S_k) + |e_{i,k+1}| \\ &= F_i(x_0) + S_k + |e_{i,k+1}| \\ &= F_i(x_0) + S_{k+1}.\end{aligned}$$

Finally, $T_i(x_{k+1}) = \max\{\bar{T}_i(x_{k+1}), F_i(x_{k+1})\}$, and both quantities are bounded above by $F_i(x_0) + S_{k+1}$. Therefore, $T_i(x_{k+1}) \le F_i(x_0) + S_{k+1}$. This completes the induction. Since

$$S_k = \sum_{r=N+1}^{k} |e_{i,r}| \le \sum_{r=N+1}^{\infty} |e_{i,r}| = E_i,$$

we conclude that $T_i(x_k) \le F_i(x_0) + E_i$, $k \ge N$. Using Lemma 4.5, we see that the result holds for all $k \ge 0$. □

Lemma 4.7. *Suppose that each objective function is bounded below. Then, for each $i = 1,\dots,m$, sequence $\{T_i(x_k)\}$ is bounded from below.*

Proof. By the standing assumption in Section 2 that each objective function F_i is bounded from below, there exists a constant $L_i \in \mathbb{R}$ such that $F_i(x) \ge L_i$, for all $x \in \mathbb{R}^n$. Since $T_i(x_k) = \max\{\bar{T}_i(x_k), F_i(x_k)\}$, it follows that $T_i(x_k) \ge F_i(x_k) \ge L_i$, for every $k \ge 0$. Hence $\{T_i(x_k)\}$ is bounded from below. □

Lemma 4.8. *Suppose that Assumption 4.3 holds. Let $\{x_k\}$ be generated by Algorithm 1. Then, for every $k \geq 0$ and every $i = 1, \ldots, m$, $T_i(x_k) - F_i(x_{k+1}) \geq c\,|\phi(x_k)|$, where $c = \min\{\mu_1, \sigma\alpha_{\min}\} > 0$.*

Proof. The proof is split into two cases.

Case 1. The trial step is accepted.

Note that $r_i^k \geq \mu_1$, $i = 1, \ldots, m$. Hence,

$$T_i(x_k) - F_i(x_{k+1}) = T_i(x_k) - F_i(x_k + d_k) = r_i^k \big(m_i^k(0) - m_i^k(d_k)\big).$$

Using $r_i^k \geq \mu_1$, Lemma 4.3, $m_i^k(0) - m_i^k(d_k) \geq |\phi(x_k)|$, we obtain $T_i(x_k) - F_i(x_{k+1}) \geq \mu_1 |\phi(x_k)|$.

Case 2. The trial step is rejected.

In this case, $x_{k+1} = x_k + \alpha_k d_k$, where $F_i(x_{k+1}) \leq T_i(x_k) + \sigma\alpha_k \nabla F_i(x_k)^\top d_k$. Therefore, $T_i(x_k) - F_i(x_{k+1}) \geq -\sigma\alpha_k \nabla F_i(x_k)^\top d_k$. Since Lemma 4.1 gives $\nabla F_i(x_k)^\top d_k \leq \phi(x_k) < 0$, we have

$$-\nabla F_i(x_k)^\top d_k \geq -\phi(x_k) = |\phi(x_k)|.$$

Hence, $T_i(x_k) - F_i(x_{k+1}) \geq \sigma\alpha_k |\phi(x_k)|$. Finally, by Assumption 4.3, $\alpha_k \geq \alpha_{\min}$, which implies $T_i(x_k) - F_i(x_{k+1}) \geq \sigma\alpha_{\min} |\phi(x_k)|$. Combining the two cases, we have $T_i(x_k) - F_i(x_{k+1}) \geq \min\{\mu_1, \sigma\alpha_{\min}\} |\phi(x_k)|$. The proof is complete. □

Lemma 4.9. *Suppose that Assumptions 4.4 and 4.3 hold. Let $\{x_k\}$ be generated by Algorithm 1. Then, for every $k \geq N$ and every $i = 1, \ldots, m$, $T_i(x_{k+1}) \leq T_i(x_k) - (1 - \eta_k) c\,|\phi(x_k)| + |e_{i,k+1}|$, where $c = \min\{\mu_1, \sigma\alpha_{\min}\} > 0$.*

Proof. By Lemma 4.8, one has $F_i(x_{k+1}) \leq T_i(x_k) - c|\phi(x_k)|$. Since $\bar{T}_i(x_{k+1}) = (1 - \eta_k) F_i(x_{k+1}) + \eta_k \bar{T}_i(x_k) + e_{i,k+1}$, it follows that

$$\bar{T}_i(x_{k+1}) \leq (1 - \eta_k)\big(T_i(x_k) - c|\phi(x_k)|\big) + \eta_k \bar{T}_i(x_k) + e_{i,k+1}.$$

Since $\bar{T}_i(x_k) \leq T_i(x_k)$, and $e_{i,k+1} \leq |e_{i,k+1}|$, we obtain

$$\bar{T}_i(x_{k+1}) \leq T_i(x_k) - (1 - \eta_k) c|\phi(x_k)| + |e_{i,k+1}|.$$

Moreover,

$$F_i(x_{k+1}) \leq T_i(x_k) - c|\phi(x_k)| \leq T_i(x_k) - (1 - \eta_k) c|\phi(x_k)| + |e_{i,k+1}|,$$

due to $0 \leq \eta_k < 1$ and $|e_{i,k+1}| \geq 0$. Therefore,

$$T_i(x_{k+1}) = \max\{\bar{T}_i(x_{k+1}), F_i(x_{k+1})\} \leq T_i(x_k) - (1 - \eta_k) c|\phi(x_k)| + |e_{i,k+1}|,$$

which completes the proof. □

Lemma 4.10. *Suppose that Assumptions 4.1–4.5 hold. Then $\sum_{k=N}^{\infty} |\phi(x_k)| < \infty$. Consequently, $\lim_{k\to\infty} \phi(x_k) = 0$.*

Proof. By Lemma 4.9, one has

$$T_i(x_{k+1}) \leq T_i(x_k) - (1 - \eta_k) c|\phi(x_k)| + |e_{i,k+1}|,$$

where $c = \min\{\mu_1, \sigma\alpha_{\min}\} > 0$. Hence, $(1 - \eta_k) c|\phi(x_k)| \leq T_i(x_k) - T_i(x_{k+1}) + |e_{i,k+1}|$. Since $0 \leq \eta_k \leq \eta_{\max} < 1$, we have $(1 - \eta_{\max}) c|\phi(x_k)| \leq T_i(x_k) - T_i(x_{k+1}) + |e_{i,k+1}|$. Summing the above inequality from $k = N$ to M gives

$$(1 - \eta_{\max}) c \sum_{k=N}^{M} |\phi(x_k)| \leq \sum_{k=N}^{M} (T_i(x_k) - T_i(x_{k+1})) + \sum_{k=N}^{M} |e_{i,k+1}|.$$

The first sum telescopes to $T_i(x_N) - T_i(x_{M+1})$. It follows from Lemma 4.7 that $T_i(x_k) \geq L_i$, for all k. Thus $T_i(x_N) - T_i(x_{M+1}) \leq T_i(x_N) - L_i$. Moreover, Assumption 4.5 implies $\sum_{k=N}^{\infty} |e_{i,k+1}| = E_i < \infty$. Consequently, $(1-\eta_{\max})c\sum_{k=N}^{M} |\phi(x_k)| \leq T_i(x_N) - L_i + E_i$. Letting $M \to \infty$ yields $\sum_{k=N}^{\infty} |\phi(x_k)| < \infty$. Hence, $\lim_{k\to\infty} |\phi(x_k)| = 0$. Since $\phi(x_k) \leq 0$ by Theorem 3.1, it follows that $\lim_{k\to\infty} \phi(x_k) = 0$. This completes the proof. □

Theorem 4.1. *Suppose that Assumptions 4.1–4.5 hold. Let $\{x_k\}$ be generated by Algorithm 1. Then every accumulation point of $\{x_k\}$ is a Pareto critical point of problem (2.1).*

Proof. By Lemma 4.10, $\lim_{k\to\infty} \phi(x_k) = 0$. Since $\{x_k\} \subset L_E(x_0)$, by Lemma 4.6 and Assumption 4.1, the sequence $\{x_k\}$ is contained in a bounded set. Hence every subsequence possesses a convergent subsequence. Let $\{x_{k_j}\}$ be a subsequence converging to an accumulation point x^*. Since $\phi(\cdot)$ is continuous by Theorem 3.1, $\phi(x^*) = \lim_{j\to\infty} \phi(x_{k_j}) = 0$. Again, by Theorem 3.1, we have that $\phi(x) = 0$ is equivalent to x being a Pareto critical point. Therefore x^* is a Pareto critical point. This completes the proof. □

5. Numerical Experiments

This section evaluates the numerical performance of the proposed trust-region algorithm with the proposed T_k nonmonotone strategy for solving unconstrained multiobjective optimization problems. The experiments are designed to assess the efficiency, robustness, and solution quality of the proposed method on a collection of benchmark problems and to compare its performance with existing trust-region algorithms reported in the literature.

For comparison, we consider three established trust-region based multi-objective methods: MMNMOTR [16], MNMOTR [29], and CMOTR [5]. These algorithms represent well-studied approaches within the trust-region framework and provide a meaningful basis for assessing the practical performance of the proposed strategy.

5.1. **Experimental setup.** All numerical experiments were carried out in MATLAB R2025b on a desktop computer equipped with an Intel Core i5-12th generation processor and 16 GB RAM running Windows 11. All algorithms were executed under the same computational environment.

The proposed algorithm was tested on a collection of 73 unconstrained multiobjective optimization problems including their variants shown in Table 1. These benchmark problems contain different characteristics, including convex and nonconvex Pareto fronts, disconnected fronts, multimodal landscapes, and different numbers of objectives and decision variables. In some of the problems, we tookt heir different variants by changing number of variables and number of objective functions. All problems are described in Table 1. For each test problem, 100 initial points were generated uniformly at random from $[x_L, x_U]^n$, and the algorithm was executed independently from each starting point. The reported values are the averages of the number of iterations, function evaluations, gradient evaluations, and CPU time over these 100 runs. Since all test problems are subject to box constraints of the form $x_L \leq x \leq x_U$, the feasibility of the trial point was enforced by incorporating the bound constraints into the trust-region subproblem. Consequently, the search direction d_k was required to satisfy $x_L \leq x_k + d_k \leq x_U$. Accordingly, instead of solving subproblem (3.2), the search direction d_k at the kth iteration

was computed by solving the following bound-constrained quadratic subproblem:

$$\left.\begin{aligned} \min \quad & t \\ \text{subject to} \quad & \nabla F_i(x_k)^\top d + \frac{1}{2} d^\top \nabla^2 F_i(x_k) d - t \le 0, \quad i = 1,2,\ldots,m, \\ & \nabla F_i(x_k)^\top d - t \le 0, \quad i = 1,2,\ldots,m, \\ & x_L - x_k \le d \le x_U - x_k, \\ & \|d\| \le \Delta_k, \\ & (t,d) \in \mathbb{R} \times \mathbb{R}^n. \end{aligned}\right\} \tag{5.1}$$

The trust-region subproblem (5.1) was solved by using the MATLAB function *fmincon* for all algorithms to ensure a consistent implementation and a fair comparison of computational performance. Since the subproblem consists of nonlinear objective and constraint functions, *fmincon* was employed as a general nonlinear programming solver.

5.1.1. *Notations used in Table 1.* In the following Table 1, we use the following notations:

- Problem: Name of test problem,
- Ref: References for the test problem,
- n: dimension of each test problem,
- m: number of objective functions in the test problem,
- x_L: lower bound of each variable,
- x_U: upper bound of each variable,
- Iter: average number of iterations consumed by the algorithm to solve the test problem,
- F-evals: average of the total number of function evaluations performed in solving the test problem,
- Grad-evals: average of the total number of gradient evaluations performed in solving the test problem,
- Time: average of total computation time in seconds for solving the test problem.

5.1.2. *Stopping criteria and parameters' values.* The algorithm was terminated whenever any one of the following conditions was satisfied:

- $|\phi(x_k)| < \varepsilon = 10^{-6}$, indicating that the first-order optimality condition was satisfied;
- $\|d_k\| < 10^{-4}$, indicating that the computed search direction was sufficiently small; or
- the number of main iterations reached the prescribed maximum of 199.

Unless otherwise specified, the parameters were fixed throughout all experiments as $\mu_1 = 0.05$, $\mu_2 = 0.9$, $\gamma_1 = 0.25$, $\gamma_2 = 2.5$, $\sigma = 0.25$, $\zeta = 0.5$, $N = 5$, $\eta_k = 0.5$, and the trust-region radius was initialized with $\Delta_0 = 1$, $\Delta_{\min} = 10^{-4}$, and $\Delta_{\max} = 100$. These parameter values were kept unchanged for all test problems.

5.2. **Numerical results.** The computational performance of the proposed algorithm is summarized in Tables 1. For each test problem, the reported quantities include the number of iterations (ITER), the number of objective function evaluations (FE), the number of gradient evaluations (GE), and the CPU time (TIME). Since the initial point is randomly generated within the prescribed variable bounds, each problem was solved independently for 100 runs, and the reported values are the averages across all runs.

TABLE 1. Average computational results of the proposed T_k-NMTR algorithm and the competing trust-region methods on benchmark multiobjective optimization problems.

S.N.	Problem	Ref	n	m	x_L	x_U	Method	Iter	F-evals	Grad-evals	Time(sec.)
1	BK1	[4]	2	2	0	5	T_k-NMTR	1.2	4.8	4.4	0.26784636
							MMNMOTR	2.6	10.4	7.2	0.31012814
							MNMOTR	2.8	11.2	7.6	0.30374874
							CMOTR	1.8	7.2	5.6	0.16982358
2	DD1a	[18]	5	2	0	2	T_k-NMTR	4.2	17.6	10.4	0.8341688
							MMNMOTR	6	24	14	0.9765189
							MNMOTR	13	52	28	2.02237996
							CMOTR	11.8	47.2	25.6	1.7824751
3	Deb41	[10]	2	2	0	1	T_k-NMTR	3	15.4	8	0.31071556
							MMNMOTR	30.2	120.8	62.4	4.06972682
							MNMOTR	18.8	75.2	39.6	2.07311854
							CMOTR	3.6	14.4	9.2	0.3695704
4	Deb513	[10]	2	2	0	1	T_k-NMTR	4	16	10	0.40169038
							MMNMOTR	7	28	16	0.71422726
							MNMOTR	10.6	42.4	23.2	1.01948744
							CMOTR	5	20	12	0.47943672
5	DIMP1	[8]	5	2	0	1	T_k-NMTR	7.4	32	16.8	1.27053164
							MMNMOTR	6	24	14	0.94231478
							MNMOTR	5.8	23.2	13.6	0.94790452
							CMOTR	4.2	16.8	10.4	0.57751548
6	DIMP1	[8]	10	2	0	1	T_k-NMTR	1.6	6.4	5.2	0.38278502
							MMNMOTR	6.4	25.6	14.8	1.59680886
							MNMOTR	2.4	9.6	6.8	0.55390496
							CMOTR	7.6	30.4	17.2	1.71864286
7	DPAM3	[9]	2	2	0	1	T_k-NMTR	7.2	31.8	16.4	1.00304746
							MMNMOTR	11	44	24	1.39011416
							MNMOTR	14.8	59.2	31.6	1.7004103
							CMOTR	8.6	34.4	19.2	1.06184716
8	DPAM3	[9]	5	2	0	1	T_k-NMTR	13.4	53.6	28.8	2.9686666
							MMNMOTR	17.4	77	36.8	4.7679273
							MNMOTR	22.2	88.8	46.4	4.73669064
							CMOTR	20	80	42	5.03750092
9	DPAM3	[9]	10	2	0	1	T_k-NMTR	22.4	95.4	46.8	11.07617272
							MMNMOTR	24.2	96.8	50.4	12.3760576
							MNMOTR	30.4	121.6	62.8	20.01335694
							CMOTR	27.2	108.8	56.4	16.52409816
10	DPAM3	[9]	20	2	0	1	T_k-NMTR	33.8	147.2	69.6	79.0420506

Table 1 continued from previous page

S.N.	Problem	Ref	n	m	x_L	x_U	Method	Iter	F-evals	Grad-evals	Time(sec.)
							MMNMOTR	61	244	124	96.45588194
							MNMOTR	103.6	414.4	209.2	161.3118961
							CMOTR	45.6	182.4	93.2	99.86356172
11	DTLZ1	[11]	6	2	0	1	T_k-NMTR	8	49.2	18	2.43531814
							MMNMOTR	15.8	63.2	33.6	4.62144316
							MNMOTR	10.6	42.4	23.2	2.69453784
							CMOTR	46.8	187.2	95.6	18.9300609
12	DTLZ1	[11]	10	2	0	1	T_k-NMTR	15.8	95	33.6	6.60348934
							MMNMOTR	67	268	136	21.33053364
							MNMOTR	49.8	199.2	101.6	20.85762808
							CMOTR	122.4	489.6	246.8	55.98887888
13	DTLZ1	[11]	7	3	0	1	T_k-NMTR	7	64.2	24	2.47040698
							MMNMOTR	12.6	75.6	40.8	4.4303648
							MNMOTR	15.8	94.8	50.4	5.30724002
							CMOTR	12.6	75.6	40.8	4.34732896
14	DTLZ1	[11]	10	3	0	1	T_k-NMTR	12.2	116.4	41.4	5.65407406
							MMNMOTR	95	570	288	48.84288236
							MNMOTR	12.8	73.2	39.6	6.71554782
							CMOTR	87.2	523.2	264.6	52.22956718
15	DTLZ1	[11]	9	5	0	1	T_k-NMTR	10	141.4	55	6.27536688
							MMNMOTR	53.2	532	271	41.35059406
							MNMOTR	10.2	102	56	8.59197228
							CMOTR	46.6	466	238	34.55245824
16	DTLZ1	[11]	15	5	0	1	T_k-NMTR	13.6	192	73	15.98412928
							MMNMOTR	24	240	125	25.26793124
							MNMOTR	129	1290	650	138.0959957
							CMOTR	52.8	528	269	56.66297416
17	DTLZ2	[11]	11	2	0	1	T_k-NMTR	5	22.4	12	1.38213344
							MMNMOTR	11.6	46.4	25.2	3.14899366
							MNMOTR	11.6	46.4	25.2	3.1429054
							CMOTR	5.2	20.8	12.4	1.38324616
18	DTLZ2	[11]	20	2	0	1	T_k-NMTR	6	26	14	3.17385246
							MMNMOTR	11	44	24	5.59731258
							MNMOTR	11.8	47.2	25.6	5.83536422
							CMOTR	7.2	28.8	16.4	3.50847656
19	DTLZ2	[11]	12	3	0	1	T_k-NMTR	6	39	21	2.46671698
							MMNMOTR	11	66	36	4.81875994
							MNMOTR	19.2	115.2	60.6	8.02308346
							CMOTR	6.8	40.8	23.4	2.63149778
20	DTLZ2	[11]	20	3	0	1	T_k-NMTR	13.8	88.2	44.4	10.4610128

Table 1 continued from previous page

S.N.	Problem	Ref	n	m	x_L	x_U	Method	Iter	F-evals	Grad-evals	Time(sec.)
							MMNMOTR	12.2	73.2	39.6	9.00052468
							MNMOTR	54.6	327.6	166.8	44.70162626
							CMOTR	6.6	39.6	22.8	4.43017232
21	DTLZ2	[11]	20	5	0	1	T_k-NMTR	5.8	66	34	7.373646
							MMNMOTR	48.6	486	248	67.472229
							MNMOTR	127	1270	640	170.269462
							CMOTR	10.6	106	58	13.2937372
22	DTLZ3	[11]	11	2	0	1	T_k-NMTR	11.8	80.8	25.6	5.54768588
							MMNMOTR	103	412	208	48.32573242
							MNMOTR	124.8	499.2	251.6	57.54849486
							CMOTR	87.2	348.8	176.4	41.83320798
23	DTLZ3	[11]	20	2	0	1	T_k-NMTR	17.2	113.8	36.4	12.91922086
							MMNMOTR	199	796	400	133.36373
							MNMOTR	199	796	400	130.7165944
							CMOTR	162	648	326	108.2828143
24	DTLZ3	[11]	12	3	0	1	T_k-NMTR	13	131.4	48	7.97235484
							MMNMOTR	126.8	760.8	383.4	79.00799306
							MNMOTR	161.4	968.4	487.2	99.48801394
							CMOTR	14	84	45	8.12927724
25	DTLZ3	[11]	20	3	0	1	T_k-NMTR	19.2	215.6	60.6	21.40823754
							MMNMOTR	162.4	974.4	490.2	149.3880572
							MNMOTR	199	1194	600	178.0839774
							CMOTR	160.8	964.8	485.4	150.4709804
26	DTLZ3	[11]	14	5	0	1	T_k-NMTR	19.8	336.4	104	26.17057884
							MMNMOTR	87.8	878	444	91.84856284
							MNMOTR	55.6	556	283	56.02626698
							CMOTR	132.6	1326	668	145.4393001
27	DTLZ3	[11]	20	5	0	1	T_k-NMTR	69	3147.2	350	172.9167776
							MMNMOTR	140.8	1408	709	213.1214888
							MNMOTR	199	1990	1000	296.967833
							CMOTR	199	1990	1000	328.9386102
28	DTLZ4	[11]	11	2	0	1	T_k-NMTR	8.6	40.2	19.2	2.58657696
							MMNMOTR	19.2	76.8	40.4	5.74187496
							MNMOTR	17.8	71.2	37.6	4.78695272
							CMOTR	8.6	34.4	19.2	2.23369462
29	DTLZ4	[11]	20	2	0	1	T_k-NMTR	10	46.6	22	5.31192038
							MMNMOTR	89.4	357.6	180.8	56.16929128
							MNMOTR	16.4	65.6	34.8	8.1632581
							CMOTR	7	28	16	3.3308986
30	DTLZ4	[11]	12	3	0	1	T_k-NMTR	7.8	46.8	26.4	3.69560996

Table 1 continued from previous page

S.N.	Problem	Ref	n	m	x_L	x_U	Method	Iter	F-evals	Grad-evals	Time(sec.)
							MMNMOTR	8.2	56.8	27.6	4.03838036
							MNMOTR	11.8	70.8	38.4	5.34872018
							CMOTR	15.4	92.4	49.2	6.72523654
31	DTLZ4	[11]	12	3	0	1	T_k-NMTR	6.2	37.2	21.6	3.40826562
							MMNMOTR	10.8	96.6	35.4	5.79134872
							MNMOTR	19.4	116.4	61.2	8.29048164
							CMOTR	12.4	74.4	40.2	4.8043434
32	DTLZ4	[11]	20	3	0	1	T_k-NMTR	16.6	159.4	52.8	12.27732368
							MMNMOTR	133.4	800.4	403.2	111.0271826
							MNMOTR	11.6	69.6	37.8	8.45864284
							CMOTR	33.8	202.8	104.4	24.79694182
33	DTLZ4	[11]	14	5	0	1	T_k-NMTR	15	289.6	80	16.1002089
							MMNMOTR	5	50	30	4.83069606
							MNMOTR	15.6	156	83	14.48051874
							CMOTR	30.8	308	159	30.30213402
34	DTLZ4	[11]	20	5	0	1	T_k-NMTR	27.8	591.6	144	46.3008149
							MMNMOTR	44.8	448	229	69.03048982
							MNMOTR	15	150	80	21.1109746
							CMOTR	26.8	268	139	39.57850634
35	DTLZ7	[11]	7	3	0	1	T_k-NMTR	24	145.5	75	7.07062925
							MMNMOTR	16	96	51	4.49214765
							MNMOTR	25	150	78	5.99813185
							CMOTR	24	144	75	5.809741
36	DTLZ7	[11]	10	3	0	1	T_k-NMTR	33	211.5	102	11.4040288
							MMNMOTR	18.5	111	58.5	6.1792972
							MNMOTR	35.5	213	109.5	11.01444985
							CMOTR	34	204	105	11.28729775
37	DTLZ7	[11]	12	3	0	1	T_k-NMTR	42	268.5	129	17.8249835
							MMNMOTR	17	102	54	6.43799595
							MNMOTR	40	240	123	14.16102455
							CMOTR	41.5	249	127.5	16.4579287
38	DTLZ7	[11]	20	3	0	1	T_k-NMTR	66	396	201	49.01834155
							MMNMOTR	199	1194	600	178.4467157
							MNMOTR	67	414	204	59.9516372
							CMOTR	66.5	399	202.5	62.02131985
39	Far1	[8]	2	2	-1	1	T_k-NMTR	5.5	23.5	13	0.69550865
							MMNMOTR	26	104	54	3.4021246
							MNMOTR	11.5	46	25	1.29911395
							CMOTR	12	48	26	1.2971291
40	Fonseca	[15]	2	2	-4	4	T_k-NMTR	5.5	23.5	13	0.55221595

Table 1 continued from previous page

S.N.	Problem	Ref	n	m	x_L	x_U	Method	Iter	F-evals	Grad-evals	Time(sec.)
							MMNMOTR	9.5	38	21	0.9630815
							MNMOTR	13	52	28	1.3964073
							CMOTR	10.5	42	23	1.05152065
41	IKK1	[21]	2	3	0	20	T_k-NMTR	11	66	36	1.3922432
							MMNMOTR	6	36	21	0.9017044
							MNMOTR	11.5	69	37.5	1.36963235
							CMOTR	12.5	75	40.5	1.3890794
42	JOS4	[21]	2	2	0	1	T_k-NMTR	6	30	14	0.7296292
							MMNMOTR	8	32	18	0.851818
							MNMOTR	8	32	18	0.7862004
							CMOTR	4.5	18	11	0.45292745
43	JOS4	[21]	5	2	0	1	T_k-NMTR	21	91.5	44	3.5912271
							MMNMOTR	71	284	144	16.06507885
							MNMOTR	36.5	146	75	6.92820825
							CMOTR	83	332	168	15.8082195
44	JOS4	[21]	10	2	0	1	T_k-NMTR	27.5	124.5	57	9.2697761
							MMNMOTR	57.5	230	117	20.52539845
							MNMOTR	112	448	226	34.70106595
							CMOTR	159	636	320	53.6904648
45	LSMOP1	[6]	100	2	0	1	T_k-NMTR	2.5	10	7	28.4893518
							MMNMOTR	101.5	406	205	1136.797152
							MNMOTR	11.5	46	25	126.4472254
							CMOTR	5.5	22	13	64.1008728
46	LSMOP2	[6]	100	2	0	1	T_k-NMTR	19.5	133	41	757.1104977
							MMNMOTR	105	420	212	2171.840469
							MNMOTR	199	796	400	4123.871095
							CMOTR	5	20	12	201.5358788
47	LSMOP2	[6]	10	2	0	1	T_k-NMTR	10	54.5	22	3.30127905
							MMNMOTR	12.5	50	27	3.56965455
							MNMOTR	14.5	58	31	3.9120628
							CMOTR	43.5	174	89	11.87972255
48	LSMOP2	[6]	20	2	0	1	T_k-NMTR	57	350.5	116	41.5840648
							MMNMOTR	13	52	28	7.48369085
							MNMOTR	10.5	42	23	6.19502955
							CMOTR	6.5	26	15	4.52530335
49	LSMOP2	[6]	10	3	0	1	T_k-NMTR	33	215	102	11.5031021
							MMNMOTR	16.5	99	52.5	5.9473256
							MNMOTR	26.5	159	82.5	8.8287303
							CMOTR	80	480	243	27.74722425
50	LSMOP2	[6]	20	3	0	1	T_k-NMTR	11.5	137	37.5	11.52538545

Table 1 continued from previous page

S.N.	Problem	Ref	n	m	x_L	x_U	Method	Iter	F-evals	Grad-evals	Time(sec.)
							MMNMOTR	107	642	324	98.1257216
							MNMOTR	66.5	399	202.5	55.89271495
							CMOTR	104	624	315	107.7800948
51	LSMOP2	[6]	10	5	0	1	T_k-NMTR	14.5	255	77.5	8.93780765
							MMNMOTR	14.5	145	77.5	8.0844106
							MNMOTR	36.5	365	187.5	20.34654645
							CMOTR	109.5	1095	552.5	56.2910373
52	LSMOP2	[6]	20	5	0	1	T_k-NMTR	87	1090.5	440	114.4739238
							MMNMOTR	105.5	1055	532.5	141.6907369
							MNMOTR	95	950	480	121.8378207
							CMOTR	135	1350	680	184.1346051
53	LSMOP3	[6]	20	2	0	1	T_k-NMTR	23.5	100	49	15.2944913
							MMNMOTR	29.5	118	61	14.24033585
							MNMOTR	37	148	76	17.5232893
							CMOTR	24	96	50	15.2066864
54	LSMOP3	[6]	5	2	0	1	T_k-NMTR	47	216.5	96	103.7190791
							MMNMOTR	110.5	442	223	180.8535257
							MNMOTR	51	204	104	77.82111645
							CMOTR	52.5	210	107	112.7092758
55	LSMOP4	[6]	10	2	0	1	T_k-NMTR	4.5	24	11	1.03006625
							MMNMOTR	8	32	18	1.9944811
							MNMOTR	12	48	26	3.4073672
							CMOTR	11.5	46	25	2.758625
56	LSMOP4	[6]	5	3	0	1	T_k-NMTR	7	42.5	16	2.9897435
							MMNMOTR	107.5	430	217	32.92142205
							MNMOTR	103.5	414	209	43.1821267
							CMOTR	104.5	418	211	47.0467736
57	LSMOP4	[6]	10	3	0	1	T_k-NMTR	9	87	30	3.0529001
							MMNMOTR	10.5	63	34.5	3.7389963
							MNMOTR	12	72	39	4.84944115
							CMOTR	10	60	33	3.3823796
58	LSMOP4	[6]	5	5	0	1	T_k-NMTR	9.5	83	31.5	4.87145885
							MMNMOTR	106	636	321	56.26813235
							MNMOTR	104.5	627	316.5	53.73202445
							CMOTR	199	1194	600	108.7034359
59	LSMOP4	[6]	10	5	0	1	T_k-NMTR	6	60	35	1.98340355
							MMNMOTR	9.5	95	52.5	3.1310401
							MNMOTR	103.5	1035	522.5	56.4365408
							CMOTR	7	78	40	2.30708965
60	MHHM1	[20]	1	3	0	1	T_k-NMTR	31	195	96	3.82289615

Table 1 continued from previous page

S.N.	Problem	Ref	n	m	x_L	x_U	Method	Iter	F-evals	Grad-evals	Time(sec.)
							MMNMOTR	12	72	39	1.520161
							MNMOTR	19	114	60	1.9572493
							CMOTR	31	186	96	2.82142205
61	T2	[31]	2	2	0	1	T_k-NMTR	5	24	12	0.63608545
							MMNMOTR	12.5	50	27	1.37467735
							MNMOTR	23.5	94	49	2.3592916
							CMOTR	4	16	10	0.4005302
62	T5	[31]	2	2	0	30	T_k-NMTR	32.5	130	67	3.6613753
							MMNMOTR	16.5	66	35	2.0576629
							MNMOTR	19.5	78	41	2.10191785
							CMOTR	23	92	48	2.255917
63	T8	[31]	3	3	0.1	10	T_k-NMTR	26	157.5	81	3.92201965
							MMNMOTR	14.5	87	46.5	2.3947123
							MNMOTR	24	144	75	3.8217406
							CMOTR	27	162	84	3.75257745
64	T8	[31]	5	3	0.1	10	T_k-NMTR	27	163.5	84	5.3256609
							MMNMOTR	15	90	48	3.333417
							MNMOTR	13	78	42	2.5345834
							CMOTR	35	210	108	6.29887495
65	T8	[31]	10	3	0.1	10	T_k-NMTR	13	79.5	42	4.04496565
							MMNMOTR	14	84	45	4.54647695
							MNMOTR	20.5	123	64.5	7.0077875
							CMOTR	35.5	213	109.5	10.50457105
66	TKLY1	[30]	4	2	-1	5	T_k-NMTR	56	235	114	7.2474601
							MMNMOTR	8	32	18	1.01374195
							MNMOTR	28	112	58	3.9375229
							CMOTR	99.5	398	201	15.01134845
67	Lovison1	[25]	2	2	-5	5	T_k-NMTR	5.1	20	12	0.666
							MMNMOTR	5.4	22.4	12.8	0.798
							MNMOTR	21.2	84.8	44.4	2.486
							CMOTR	11.8	47.2	25.6	1.187922
68	Lovison2	[25]	2	2	-5	5	T_k-NMTR	22.8	91.2	47.6	2.64
							MMNMOTR	45	189.2	92	7.199
							MNMOTR	27.4	109.6	56.8	2.918
							CMOTR	11.6	46.4	25.2	1.251077
69	Lovison3	[25]	2	2	-5	5	T_k-NMTR	10	40.8	22	1.027
							MMNMOTR	13.2	52.8	28.4	1.456
							MNMOTR	20.4	81.6	42.8	2.377
							CMOTR	15	60	32	1.534156
70	Lovison4	[25]	2	2	-5	5	T_k-NMTR	3.4	14.8	8.8	0.367

Table 1 continued from previous page

S.N.	Problem	Ref	n	m	x_L	x_U	Method	Iter	F-evals	Grad-evals	Time(sec.)
							MMNMOTR	13.2	52.8	28.4	1.483
							MNMOTR	22.2	88.8	46.4	2.316
							CMOTR	9.6	38.4	21.2	1.014767
71	MOP6	[34]	2	2	0	1	T_k-NMTR	5.4	23.6	13.6	0.728
							MMNMOTR	6.6	26.4	15.2	0.734
							MNMOTR	5.8	23.2	13.6	0.891
							CMOTR	9	36	20	1.410442
72	Jin1	[22]	2	2	0	2	T_k-NMTR	1.2	4.8	4.4	0.33684334
							MMNMOTR	2	8	6	0.5507707
							MNMOTR	2	8	6	0.5548827
							CMOTR	1.3	4.8	4.4	0.31300498
73	Jin3	[22]	2	2	0	1	T_k-NMTR	6	24	14	1.53903104
							MMNMOTR	8	32	18	2.5322383
							MNMOTR	8.4	33.6	18.8	2.3258231
							CMOTR	7.6	31.6	17.2	2.3953316

To provide an overall comparison of the competing algorithms, Table 2 summarizes the number of benchmark problems on which each method achieves the unique best performance with respect to each performance metric. A win is counted only when a method attains the strictly smallest value among all competing methods, while tied best results are reported separately in the last column.

TABLE 2. Number of strict wins obtained by each algorithm for the four performance metrics over all benchmark problems. Tied results are reported separately.

Metric	T_k-NMTR	MMNMOTR	MNMOTR	CMOTR	Ties
Iterations (ITER)	50	10	3	8	2
Function Evaluations (FE)	36	13	9	14	1
Gradient Evaluations (GE)	46	10	4	9	4
CPU Time (TIME)	46	12	4	11	0

As shown in Table 2, the proposed T_k-NMTR algorithm achieves the largest number of strict wins for all four performance metrics. In particular, it records 50 strict wins in terms of the number of iterations, substantially outperforming MMNMOTR, MNMOTR, and CMOTR, which achieve 10, 3, and 8 strict wins, respectively. A similar trend is observed for the number of gradient evaluations, where the proposed method attains 46 strict wins, indicating that the proposed nonmonotone strategy effectively reduces the computational effort required to satisfy the termination criteria.

The proposed algorithm also performs favorably with respect to the number of objective function evaluations and CPU time, obtaining 36 and 46 strict wins, respectively. These results indicate that the reduction in iteration count is accompanied by corresponding reductions in computational cost, rather than being achieved at the expense of additional function or gradient

evaluations. Consequently, the proposed T_k strategy provides an effective balance between convergence efficiency and computational overhead.

It is worth noting that only a few benchmark problems result in tied best performances, with two ties in Iterations, one tie in Function-evaluations, and four ties in Gradient-evaluations, while no ties are observed for CPU time. This indicates that the superiority of the proposed algorithm is generally unambiguous across the benchmark suite. Nevertheless, the competing methods remain competitive on several individual problems, suggesting that the effectiveness of a trust-region strategy is influenced by the underlying problem characteristics. Overall, the proposed T_k-NMTR algorithm demonstrates the most consistent numerical performance among the four methods considered.

5.3. **Performance profile analysis.** To further assess the efficiency and robustness of the competing algorithms, we employ the performance profile proposed by Dolan and Moré [12]. Performance profiles are presented with respect to the average number of iterations, average number of objective function evaluations, average number of gradient evaluations, average CPU time, Hypervolume (HV), Inverted Generational Distance Plus (IGD^+), and Purity.

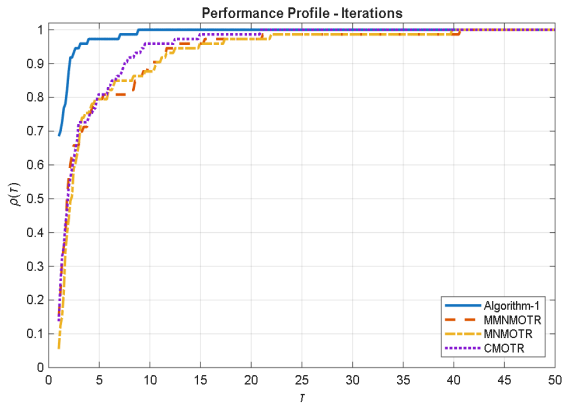
Let $\mathbb{S}$ denote the set of competing algorithms and let P denote the set of benchmark problems. Suppose that $t_{p,s}$ represents the performance measure associated with solver $s \in \mathbb{S}$ on problem $p \in P$. For computational measures (Iter, F-evals, Grad-evals, and Time), smaller values are preferred. Accordingly, the performance ratio is defined by $r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s}:s\in\mathbb{S}\}}$, where $r_{p,s} \geq 1$ for all $p \in P$ and $s \in \mathbb{S}$. The corresponding performance profile of solver s is given by

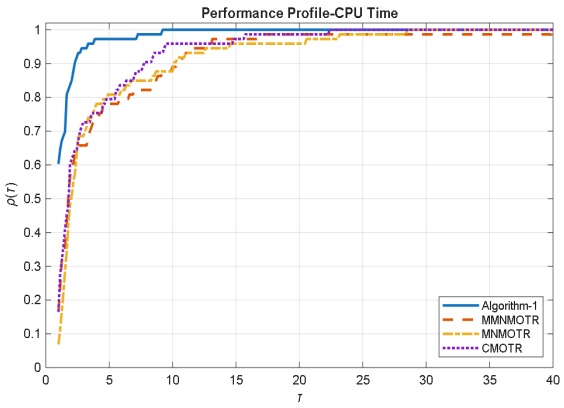
$$P_s(\tau) = \frac{\left|\left\{p \in P : r_{p,s} \leq \tau\right\}\right|}{|P|}, \qquad \tau \geq 1,$$

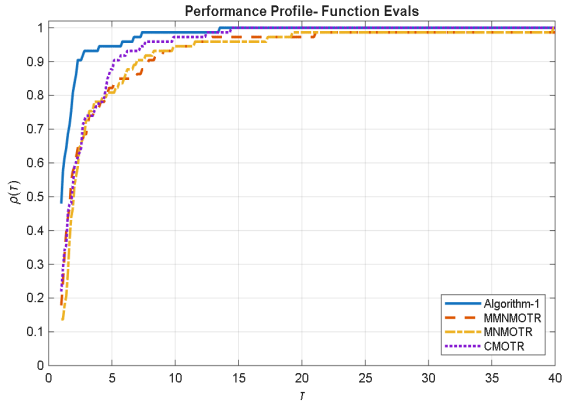
where $|P|$ denotes the total number of benchmark problems. Thus, $P_s(\tau)$ represents the probability that solver s solves a problem within a factor τ of the best-performing solver. For the quality indicators HV, IGD^+, and Purity, the same performance profile framework is employed after accounting for the optimization direction of each metric. Specifically, larger values are preferred for HV and Purity, whereas smaller values are preferred for IGD^+. In the performance profiles, larger values of $P_s(\tau)$ indicate better overall performance. In particular, the solver with the largest value of $P_s(1)$ is regarded as the most efficient, since it achieves the best performance on the largest proportion of benchmark problems. As τ increases, the algorithm with the highest performance profile is considered the most robust because it successfully solves a larger fraction of problems within a prescribed factor of the best observed performance.

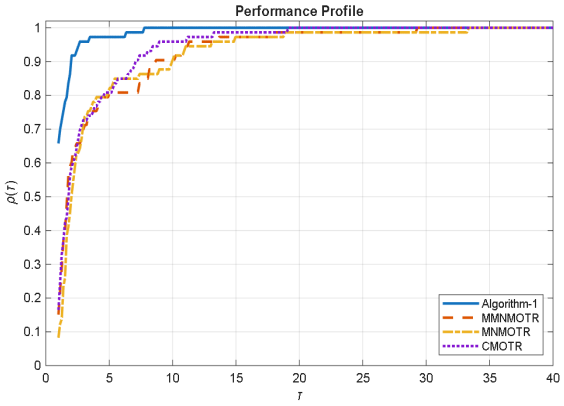
The following seven figures present the performance profiles of the proposed T_k-NMTR algorithm (Algorithm 1) together with MMNMOTR, MNMOTR, and CMOTR with respect to the average number of iterations, average number of objective function evaluations, average number of gradient evaluations, average CPU time, Hypervolume (HV), IGD^+, and Purity, respectively.

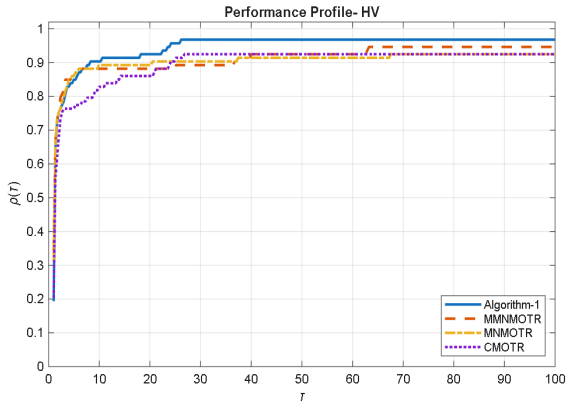
For the computational performance measures, namely iterations, function evaluations, gradient evaluations, and CPU time, the proposed T_k-NMTR algorithm consistently exhibits the best overall performance. In particular, the value of the performance profile at $\tau = 10$ is considerably larger than those of the competing algorithms, indicating that the proposed method most frequently attains the best performance over the benchmark problems. Furthermore, the performance profile of the proposed method remains above those of MMNMOTR, MNMOTR,

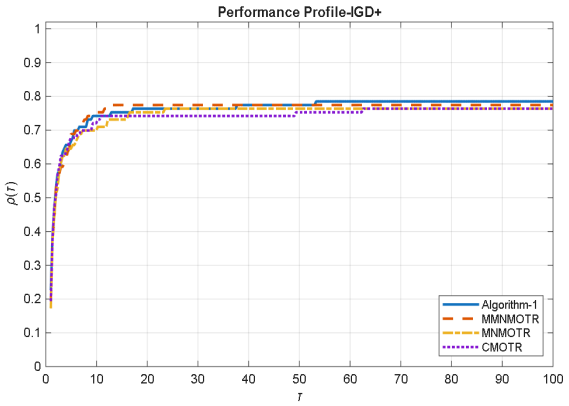

Performance Profile - Iterations
ρ(τ)
τ
Algorithm-1
MMNMOTR
MNMOTR
CMOTR

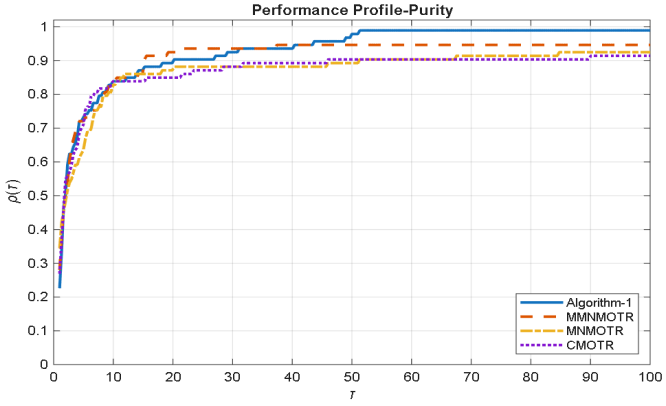

Performance Profile-CPU Time
ρ(τ)
τ
Algorithm-1
MMNMOTR
MNMOTR
CMOTR


Performance Profile- Function Evals
ρ(τ)
τ
Algorithm-1
MMNMOTR
MNMOTR
CMOTR


Performance Profile
ρ(τ)
τ
Algorithm-1
MMNMOTR
MNMOTR
CMOTR


Performance Profile- HV
ρ(τ)
τ
Algorithm-1
MMNMOTR
MNMOTR
CMOTR


Performance Profile-IGD+
ρ(τ)
τ
Algorithm-1
MMNMOTR
MNMOTR
CMOTR


Performance Profile-Purity
ρ(τ)
τ
Algorithm-1
MMNMOTR
MNMOTR
CMOTR

and CMOTR over almost the entire range of performance ratios, demonstrating superior computational efficiency as well as robustness. These observations agree with the numerical results reported in the previous tables, where the proposed method achieves the largest number of strict wins in terms of iterations, function evaluations, gradient evaluations, and CPU time.

The performance profiles based on the quality indicators HV, IGD^+, and Purity further demonstrate the effectiveness of the proposed algorithm. In the Hypervolume profile, the proposed T_k-NMTR algorithm achieves the highest probability of producing the best approximation set over most of the performance-ratio range, indicating better convergence and diversity of the obtained Pareto front. Similarly, for the IGD^+ metric, the proposed method maintains the highest performance profile throughout almost the entire interval, implying that it more frequently generates approximation sets that are closer to the true Pareto front. The Purity profile also shows that the proposed algorithm consistently outperforms the competing methods and approaches the highest probability among all solvers, suggesting that the obtained nondominated solution sets contain a larger proportion of high-quality Pareto-optimal solutions.

Overall, the performance profiles confirm the conclusions drawn from the detailed numerical results. The proposed T_k-NMTR algorithm not only reduces the computational effort required to solve unconstrained multiobjective optimization problems but also produces high-quality Pareto approximations. Consequently, the proposed nonmonotone trust-region strategy provides a favorable balance between computational efficiency, robustness, and approximation quality.

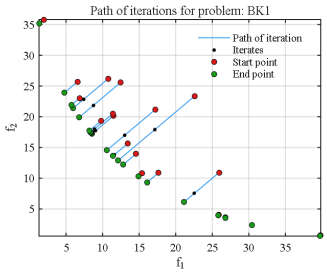
5.4. **Trajectory Behavior.** To provide a visual illustration of the convergence behavior of the proposed algorithm, Figure 1 presents the iteration paths in the objective space for several representative benchmark problems. Starting from different initial points, the iterates consistently progress toward the Pareto-optimal region and eventually converge to the final solutions. Although the trajectories vary according to the characteristics of the benchmark problems, the overall behavior remains stable, indicating that the proposed T_k-NMTR algorithm effectively guides the search process from diverse initial points while maintaining reliable convergence.

Overall, the numerical experiments demonstrate the effectiveness of the proposed T_k-NMTR algorithm for solving unconstrained multiobjective optimization problems. The detailed numerical results, summary statistics, performance profiles, and iteration-path plots consistently show that the proposed method achieves competitive or superior performance in terms of computational efficiency and solution quality. These results indicate that the proposed nonmonotone trust-region strategy provides a robust and efficient framework for computing high-quality Pareto-optimal solutions across a diverse collection of benchmark problems.

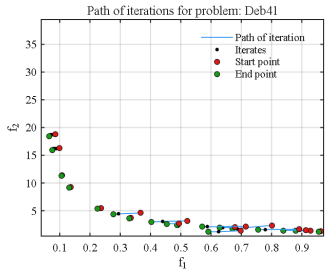
6. Conclusion

In this paper, we proposed a new trust-region algorithm for solving unconstrained multiobjective optimization problems based on a novel T_k nonmonotone strategy. The proposed approach incorporates a nonmonotone reference term into the trust-region framework, providing greater flexibility in accepting trial points while preserving the global convergence properties of the algorithm. Under standard assumptions, global convergence of the proposed method was established by proving that every accumulation point of the generated sequence satisfies the first-order Pareto criticality conditions.

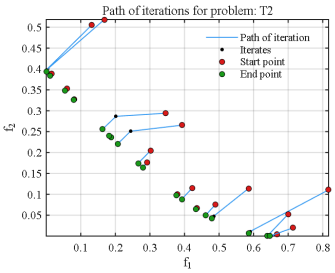
The effectiveness of the proposed algorithm was demonstrated through extensive numerical experiments on a diverse collection of unconstrained multiobjective optimization benchmark



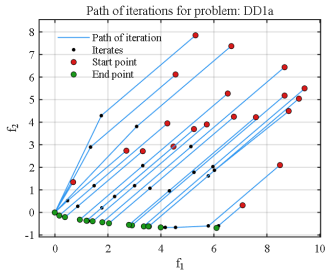
(A) Problem: BK1



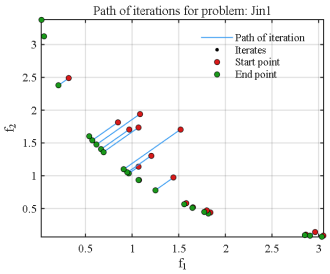
(B) Problem: Deb41



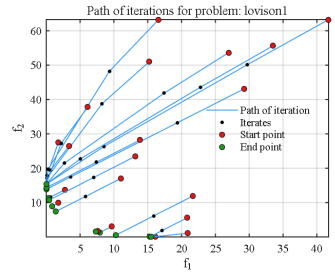
(C) Problem: T2



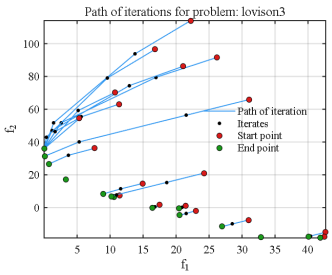
(D) Problem: DD1a



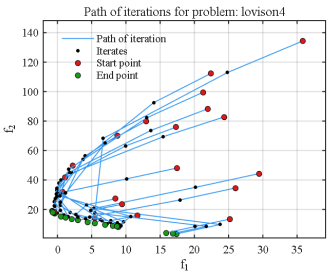
(E) Problem: Jin1



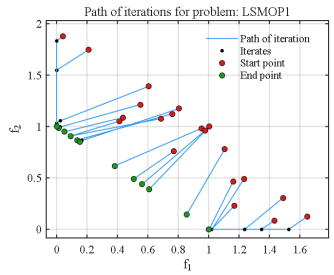
(F) Problem: Lovison1



(G) Problem: Lovison3



(H) Problem: Lovison4



(I) Problem: LSMOP1

FIGURE 1. Evolution of the iteration paths in the objective space for representative unconstrained multiobjective optimization problems. Red circles denote the initial points, black dots represent the intermediate iterates, and green circles indicate the final points obtained by the proposed T_k-NMTR algorithm.

problems. The numerical results showed that the proposed T_k-NMTR algorithm consistently performs competitively and, in many cases, outperforms the existing MMNMOTR, MNMOTR, and CMOTR algorithms. This conclusion is supported by the detailed numerical results, the summary statistics of strict wins, the performance profile analysis, and the iteration-path illustrations. In particular, the proposed algorithm achieved superior computational efficiency in terms of iterations, objective function evaluations, gradient evaluations, and CPU time, while simultaneously producing high-quality Pareto approximations as measured by the Hypervolume, IGD$^+$, and Purity indicators. Although the proposed algorithm has been developed for unconstrained multiobjective optimization problems, several interesting research directions remain. Future work will focus on extending the proposed framework to constrained multiobjective optimization problems by incorporating suitable feasibility restoration or filter techniques.

Another promising direction is the development of quasi-Newton and limited-memory variants to improve scalability for large-scale optimization problems. Furthermore, adaptive strategies for selecting the nonmonotone parameter and trust-region radius, together with applications to practical engineering and machine learning optimization problems, constitute worthwhile topics for future investigation.

Acknowledgements
The first author was supported by the National Natural Science Foundation of China (Grant No. 12271071 and Grant No. 11991024) and the Team Project of Innovation Leading Talent in Chongqing (Grant No. CQYC20210309536).

REFERENCES

[1] K. Amini, M. Ahookhosh, A hybrid of adjustable trust-region and nonmonotone algorithms for unconstrained optimization, Appl. Math. Model. 38 (2014), 2601–2612.
[2] M. Ahookhosh, S. Ghaderi, Two globally convergent nonmonotone trust-region methods for unconstrained optimization, J. Appl. Math. Comput. 50 (2016), 529–555.
[3] M. Ahookhosh, K. Amini, An efficient nonmonotone trust-region method for unconstrained optimization, Numer. Algorithms 59 (2012), 523–540.
[4] T.T. Binh, U. Korn, MOBES: A multiobjective evolution strategy for constrained optimization problems, In: Proc. 3rd Int. Conf. Genetic Algorithms (Mendel 97), Brno, Czech Republic, pp. 176–182, 1997.
[5] G.A. Carrizo, P.A. Lotito, M.C. Maciel, Trust-region globalization strategy for the nonconvex unconstrained multiobjective optimization problem, Math. Program. 159 (2016), 339–369.
[6] R. Cheng, Y. Jin, M. Olhofer, B. Sendhoff, Test problems for large-scale multiobjective and many-objective optimization, IEEE Trans. Cybern. 47 (2017), 4108–4121.
[7] A.R. Conn, N.I.M. Gould, P.L. Toint, Trust Region Methods, SIAM, Philadelphia, PA, 2000.
[8] A.L. Custódio, J.F.A. Madeira, A.I.F. Vaz, L.N. Vicente, Direct multisearch for multiobjective optimization, SIAM J. Optim. 21 (2011), 1109–1140.
[9] I. Das, J.E. Dennis, Normal-Boundary Intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems, SIAM J. Optim. 8 (1998), 631–657.
[10] K. Deb, Multi-objective genetic algorithms: Problem difficulties and construction of test problems, Evol. Comput. 7 (1999), 205–230.
[11] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multiobjective optimization, In: A. Abraham, L. Jain, R. Goldberg (eds.) Evolutionary Multiobjective Optimization: Theoretical Advances and Applications, pp. 105–145, Springer, London, 2005.
[12] E.D. Dolan, J.J. Moré, Benchmarking optimization software with performance profiles, Math. Program. 91 (2002), 201–213.
[13] J. Fliege, B.F. Svaiter, Steepest descent methods for multicriteria optimization, Math. Methods Oper. Res. 51 (2000), 479–494.
[14] J. Fliege, L.M. Graña Drummond, B.F. Svaiter, Newton's method for multiobjective optimization, SIAM J. Optim. 20 (2009), 602–626.
[15] C.M. Fonseca, P.J. Fleming, An overview of evolutionary algorithms in multiobjective optimization, Evol. Comput. 3 (1995), 1–16.
[16] N. Ghalavand, E. Khorram, V. Morovati, Two adaptive nonmonotone trust-region algorithms for solving multiobjective optimization problems, Optimization 73 (2024), 2953–2985.
[17] M.L.N. Gonçalves, F.S. Lima, L.F. Prudente, Globally convergent Newton-type methods for multiobjective optimization, Comput. Optim. Appl. 83 (2022), 403–434.
[18] L.M. Graña Drummond, A.N. Iusem, A projected gradient method for vector optimization problems, Comput. Optim. Appl. 28 (2004), 5–29.

[19] K. Gupta, D. Ghosh, C. Tammer, X. Zhao, J. C. Yao, A quasi-newton method to solve uncertain multiobjective optimization problems with uncertainty set of finite cardinality, J. Nonlinear Var. Anal. 10 (2026), 635-660.
[20] J. Hämäläinen, K. Miettinen, P. Tarvainen, J. Toivanen, Interactive solution approach to a multiobjective optimization problem in a paper machine headbox design, J. Optim. Theory Appl. 116 (2003), 265–281.
[21] S. Huband, P. Hingston, L. Barone, L. While, A review of multiobjective test problems and a scalable test problem toolkit, IEEE Trans. Evol. Comput. 10 (2006), 477–506.
[22] Y. Jin, T. Okabe, B. Sendhoff, Adapting weighted aggregation for multiobjective evolution strategies, In: Evolutionary Multi-Criterion Optimization (EMO 2001), Lecture Notes in Computer Science, vol. 1993, pp. 96–110, Springer, Berlin, 2001.
[23] S. Kumar, N.K. Mahato, M.A.T. Ansary, D. Ghosh, Solving an uncertain quadratic multiobjective optimization problem using Newton's descent method via a robust optimization approach, J. Appl. Numer. Optim. 7 (2025), 149–177.
[24] K. Kumar, A. Singh, A. Upadhayay, D. Ghosh, An improved nonmonotone quasi-Newton method for multiobjective optimization problems, J. Nonlinear Convex Anal. 27 (2026), 119–142.
[25] A. Lovison, Singular continuation: Generating piece-wise linear approximations to Pareto sets via global analysis, J. Optim. Theory Appl. 149 (2011), 463–490.
[26] L.R. Lucambio Pérez, L.F. Prudente, Nonlinear conjugate gradient methods for vector optimization, SIAM J. Optim. 28 (2018), 2690–2720.
[27] K. Mita, E.H. Fukuda, N. Yamashita, Nonmonotone line searches for unconstrained multiobjective optimization problems, J. Global Optim. 75 (2019), 63–90.
[28] J.W. Peng, D.H. Zhong, A. Singh, A novel modified Liu-Storey nonlinear conjugate gradient method for solving vector optimization problems, Optimization (2025), 1-30.
[29] V.A. Ramirez, G.N. Sottosanto, Nonmonotone trust region algorithm for solving the unconstrained multiobjective optimization problems, Comput. Optim. Appl. 81 (2022), 769-788.
[30] H. Tamaki, H. Kita, S. Li, G. Yamazaki, A multi-objective genetic algorithm with vector-exchange mechanism, In: Proc. 5th Int. Conf. Parallel Problem Solving from Nature (PPSN V), Lect. Notes Comput. Sci., vol. 1498, pp. 511–520, Springer-Verlag, 1998.
[31] J. Thomann, G. Eichfelder, Numerical results for the multiobjective trust region algorithm MHT, Data Brief 25 (2019), 104103.
[32] J. Yahaya, P. Kumam, C. Tammer, New three-term conjugate gradient method for multiobjective optimization problems, Ann. Math. Sci. Appl. 10 (2025), 633–665.
[33] H.H. Yang, Q.H. Zhou, A trust-region method with a nonmonotone technique for nonconvex multiobjective optimization, Int. J. Appl. Math. Soft Comput. 4 (2018), 59–63.
[34] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: Empirical results, Evol. Comput. 8 (2000), 173–195.
[35] Y. Zhao, C.L. You, W. Chen, Gradient-free methods for nonsmooth nonconvex multiobjective optimization, Optim. Erud. 2 (2025), 157-164.
[36] H. Zhang, W.W. Hager, A nonmonotone line search technique and its application to unconstrained optimization, SIAM J. Optim. 14 (2004), 1043–1056.