symmetry

MDPI

Article

DTSA: Dynamic Tree-Seed Algorithm with Velocity-Driven Seed Generation and Count-Based Adaptive Strategies

Jianhua Jiang [1,2,*], Jiansheng Huang [1,2], Jiaqi Wu [1,2], Jinmeng Luo [1,2], Xi Yang [1,2] and Weihua Li [3]

[1] Center for Artificial Intelligence, Jilin University of Finance and Economics, Changchun 130117, China; huangjiansheng55@163.com (J.H.); 5221191008@s.jlufe.edu.cn (J.W.); 5221191007@s.jlufe.edu.cn (J.L.); acey_yang@163.com (X.Y.)
[2] Jilin Province Key Laboratory of Fintech, Jilin University of Finance and Economics, Changchun 130117, China
[3] School of Engineering, Computer and Mathematical Sciences, Auckland University of Technology, Auckland 1010, New Zealand; weihua.li@aut.ac.nz
* Correspondence: jjh@jlufe.edu.cn

Abstract: The Tree-Seed Algorithm (TSA) has been effective in addressing a multitude of optimization issues. However, it has faced challenges with early convergence and difficulties in managing high-dimensional, intricate optimization problems. To tackle these shortcomings, this paper introduces a TSA variant (DTSA). DTSA incorporates a suite of methodological enhancements that significantly bolster TSA's capabilities. It introduces the PSO-inspired seed generation mechanism, which draws inspiration from Particle Swarm Optimization (PSO) to integrate velocity vectors, thereby enhancing the algorithm's ability to explore and exploit solution spaces. Moreover, DTSA's adaptive velocity adaptation mechanism based on count parameters employs a counter to dynamically adjust these velocity vectors, effectively curbing the risk of premature convergence and strategically reversing vectors to evade local optima. DTSA also integrates the trees population integrated evolutionary strategy, which leverages arithmetic crossover and natural selection to bolster population diversity, accelerate convergence, and improve solution accuracy. Through experimental validation on the IEEE CEC 2014 benchmark functions, DTSA has demonstrated its enhanced performance, outperforming recent TSA variants like STSA, EST-TSA, fb-TSA, and MTSA, as well as established benchmark algorithms such as GWO, PSO, BOA, GA, and RSA. In addition, the study analyzed the best value, mean, and standard deviation to demonstrate the algorithm's efficiency and stability in handling complex optimization issues, and DTSA's robustness and efficiency are proven through its successful application in five complex, constrained engineering scenarios, demonstrating its superiority over the traditional TSA by dynamically optimizing solutions and overcoming inherent limitations.

Keywords: swarm intelligence; Tree-Seed Algorithm; PSO-inspired seed generation mechanism; engineering optimization problems

Citation: Jiang, J.; Huang, J.; Wu, J.; Luo, J.; Yang, X.; Li, W. DTSA: Dynamic Tree-Seed Algorithm with Velocity-Driven Seed Generation and Count-Based Adaptive Strategies. *Symmetry* **2024**, *16*, 795. https://doi.org/10.3390/sym16070795

Academic Editors: Jianchao Bai, Qiyu Jin and Yuchao Tang

Received: 26 May 2024
Revised: 18 June 2024
Accepted: 19 June 2024
Published: 25 June 2024

1. Introduction

An optimization problem involves the pursuit of the optimal or an approximate optimal solution within prescribed constraints [1]. This typically necessitates identifying variable values that either maximize or minimize a specific objective function [2,3]. Such problems are fundamental in scientific inquiry, engineering, and various other disciplines, driving advancements in diverse fields through systematic problem-solving approaches [4–6]. However, when solving optimization problems in the real world, we encounter countless complexities, including high computational requirements, nonlinear constraints, dynamic or noisy objective functions, etc. [7]. These challenges significantly impact the decision-making process for selecting the appropriate optimization algorithms. Against this backdrop, exact algorithms provide precise global optimum solutions. However, as the number of variables increases, their execution time grows exponentially. This

exponential growth makes them impractical for large-scale or highly complex scenarios [8]. Therefore, stochastic optimization algorithms, especially heuristic and metaheuristic algorithms within the realm of approximate solutions, emerge as practical tools for tackling complex issues [9,10].

Heuristic and metaheuristic algorithms are favored for their ability to search for optimal or near-optimal solutions within a reasonable timeframe by employing probabilistic and statistical methods [11,12]. These algorithms are particularly suited for addressing complex problems that are beyond the reach of traditional methods. On the one hand, heuristics utilize problem-specific strategies, relying on intuitive logic to make decisions that guide the search process towards promising areas of the solution space [13,14]. On the other hand, metaheuristics provide a higher level of abstraction, offering a framework adaptable to various optimization problems without the need for customization to a specific issue [15]. This shift towards heuristic and metaheuristic algorithms represents a natural response to the limitations of exact algorithms, underscoring their significance in solving the complex optimization challenges characteristic of many real-world applications [6,16,17].

As depicted in Figure 1, metaheuristic algorithms can be broadly categorized as non-nature-inspired or nature-inspired. Examples of non-nature-inspired algorithms include Tabu Search (TS) [18,19], Iterated Local Search (ILS) [20,21], and Adaptive Dimensional Search (ADS) [22], many algorithms draw inspiration from natural phenomena. Nature-inspired algorithms, such as Differential Evolution (DE) [23,24], Particle Swarm Optimization (PSO) [25,26], Artificial Bee Colony (ABC) [27,28], Krill Herd (KH) [29,30], and Gravitational Search Algorithm (GSA) [31,32], are valued for their versatility and simplicity in tackling complex optimization challenges [23]. However, it is essential to note that while some algorithms excel in addressing specific problems, their effectiveness may vary across different or intricate scenarios [33].

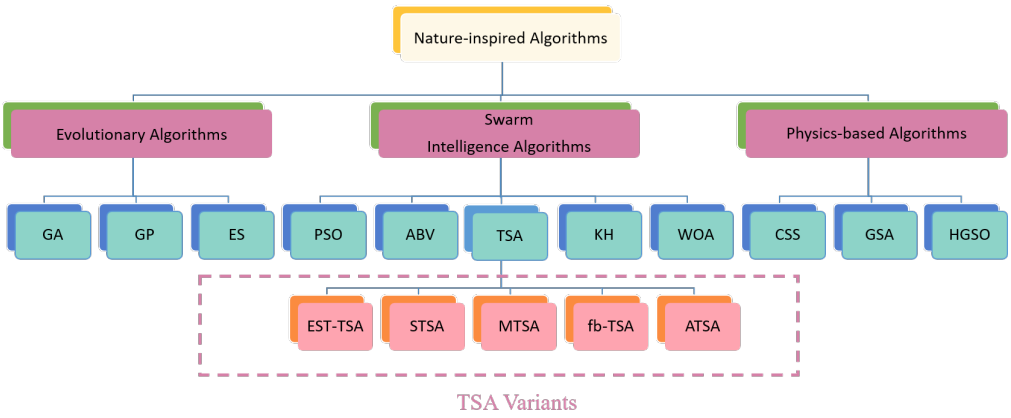


Figure 1. The classification of nature-inspired algorithms.

Among them, the Tree-Seed Algorithm (TSA) stands out as a successful nature-inspired metaheuristic developed in 2015, drawing inspiration from the natural interplay between tree and seed evolution [34]. TSA approaches optimization problems by exploring the locations of trees and seeds, representing feasible solutions [35,36]. Notably, TSA has gained widespread adoption, surpassing some swarm intelligence algorithms in popularity due to its simpler structure, higher precision, and greater robustness [37]. Nevertheless, TSA exhibits drawbacks, including premature convergence and a susceptibility to stagnate in local optima [38]. This study aims to mitigate these issues through novel approaches, including a distinct initial design and algorithmic modifications and hybridizations.

1.1. Motivations

TSA is a population-based approach to evolution that takes its inspiration from the relationship between trees and seeds, which are spread to the ground for reproduction. TSA is simple, has few parameters, it is easy to understand the concept, and it is effective in solving continuous structure optimization problems [34]. However, the TSA has its limits. During space exploration, it is often trapped in a locally optimal solution, struggling to escape this mechanism [39]. In addition, the optimal solutions generated for each round are underutilized, resulting in a relatively single optimal position [40]. These shortcomings highlight the need to refine the TSA methodology to improve its effectiveness and extend its applicability to optimization tasks. However, it is these shortcomings that motivate us to improve it. The motivations driving this research are as follows:

- The existing seed generation mechanism in TSA lacks consideration of population attributes and may lead to seeds being generated in less favorable regions of the search space [41]. By redesigning the tree selection process, we aim to improve the quality of generated seeds and enhance the algorithm's overall performance.
- In the TSA evolutionary process, interactions solely between trees and seeds ignore potential interactions among trees, reducing population diversity [42]. To tackle this, we propose a tree population strategy to boost diversity and speed up convergence.
- Many optimization algorithms rely on simplistic initialization methods, such as uniform random sampling, which may result in a less diverse initial population [43]. By rethinking the initialization process, we aim to improve the exploration–exploitation balance and enhance the algorithm's robustness across various optimization scenarios.

1.2. Contribution

The no free lunch (NFL) theorem holds significance within this context [44]. This study is notably driven by the explicit goal of reducing the probability of conventional TSA encountering stagnation in local optima and premature convergence. This study aims to enhance the balance between exploitation and exploration by incorporating the PSO velocity update mechanism and an improved balance mechanism into the original TSA. The primary contributions of this paper can be succinctly summarized as follows.

- PSO-Inspired seed generation mechanism: The significant advancement of this TSA variant lies in its seed generation technique, which is inspired by PSO. By utilizing velocity vectors for updating seed positions, this approach introduces a dynamic and adaptive element to the exploration–exploitation equilibrium, thereby augmenting the algorithm's efficacy in traversing the search space efficiently.
- Adaptive dynamic parameter update mechanism: The incorporation of adaptive weight (w) and constant (k) updates during the optimization process is a novel aspect. This adaptive mechanism allows the algorithm to dynamically adjust its exploration and exploitation tendencies based on the current iteration, contributing to improved convergence behavior and solution quality.
- Adaptive velocity adaptation mechanism based on count parameters: The introduction of a count-based adaptive mechanism for updating the velocity vectors contributes to the algorithm's ability to dynamically adjust its behavior during different phases of the optimization process. The count parameter influences the exploration–exploitation trade-off, allowing the algorithm to adapt its strategy as the optimization progresses.
- Population-based evolutionary strategy with information exchange: The variant integrates an evolutionary strategy characterized by dynamic partitioning of the population into distinct subpopulations based on their respective fitness values. This innovative approach incorporates a combination of grouping, crossover, and natural selection operations. Crossover events are facilitated between the superior subpopulation and a subset of the inferior subpopulation, facilitating structured information exchange. Concurrently, a natural selection operation replaces the inferior subpopulation with the superior counterpart in terms of both position and velocity. This sophisticated methodology enhances the algorithm's adaptability, facilitating the ef-

fective exploitation of promising solutions while concurrently preserving population diversity to mitigate premature convergence.

- Dynamic seeding with chaotic map: The sine chaotic map is used for generating random numbers during the initialization phase and seed production [45]. Chaotic maps can provide a better and more dynamic exploration of the search space compared to uniform random numbers. This can enhance the diversity of the seeds produced and potentially improve the algorithm's ability to escape from local optima.

2. Related Work

2.1. A Brief Introduction to TSA

The TSA, proposed by Mustafa Servet Kiran in 2015, is a swarm intelligence algorithm inspired by the relationship between trees and seeds on land [34]. TSA is designed for solving continuous optimization problems and is widely applicable in heuristic and population-based search scenarios. The key principles of TSA are outlined below:

- Tree position initialization: The initial position of each tree (T_i) is determined by randomly selecting values for each dimension (j) within the specified bounds of the search space, using Equation (1).

$$T_{i,j} = L_{j,\min} + r_{i,j} \times (H_{j,\max} - L_{j,\min}) \tag{1}$$

  where $T_{i,j}$ is the j-th dimension of the i-th tree, $L_{j,\min}$ and $H_{j,\max}$ are the lower and upper bounds of the search space for dimension j, and $r_{i,j}$ is a random number in the range of $[0, 1]$.
- Tree-seed renewal mechanism: The seed renewal mechanism involves two update formulas for generating new seeds, considering both the current tree's location and the optimal location of the entire tree population, which are calculated in Equations (2) and (3).

$$\text{Local Search Formula:} \quad S_{i,j} = T_{i,j} + \alpha_{i,j} \times (B_j - T_{r,j}) \tag{2}$$

$$\text{Global Search Formula:} \quad S_{i,j} = T_{i,j} + \alpha_{i,j} \times (T_{i,j} - T_{r,j}) \tag{3}$$

  where $S_{i,j}$ is the j-th dimension of the seed to be produced by the i-th tree, $T_{i,j}$ is the j-th dimension of the i-th tree, B_j is the j-th dimension of the best tree location obtained so far, $T_{r,j}$ is the j-th dimension of a randomly selected tree (r) from the population, and $\alpha_{i,j}$ is a scaling factor randomly generated in the range $[-1, 1]$.

2.2. Literature Review

The TSA has gained significant attention in the swarm intelligence research community. It has become widely adopted due to its simplicity, precision, and robustness in comparison to other swarm intelligence algorithms [46]. Researchers have continuously proposed variants to improve TSA's performance, with a focus on enhancing seed generation, tree migration, and expanding its application domains [35].

- Tree migration variants: The Migration Tree-Seed Algorithm (MTSA) incorporates hierarchical gravity learning and random-based migration, drawing inspiration from the Grey Wolf Optimizer [38]. This approach effectively mitigates challenges related to exploration–exploitation imbalance, local stagnation, and premature convergence. Additionally, the Triple Tree-Seed Algorithm (TriTSA) introduces triple learning-based mechanisms, amalgamating migration strategies with sine random distribution to further enhance algorithmic performance [47].
- Innovations in seed generation: Various innovations have emerged to enhance seed generation and improve the effectiveness of the optimization process. Jiang's integration of the Sine Cosine Algorithm (SCA) with TSASC introduces a novel mechanism for updating seed positions, refining weight factors to pursue optimal solutions [48]. Additionally, the Sine Tree-Seed Algorithm (STSA) dynamically adjusts seed quantity, transitioning from higher to lower counts to emphasize output bolstering during

initial search phases [42]. Other TSA variants like ITSA, incorporating an acceleration coefficient for faster updates [49], and EST-TSA, leveraging the current optimal population position for improved local search, make significant contributions [50]. Innovations such as fb-TSA [51], integrating seeds and search tendencies via feedback mechanisms [51], and LTSA, introducing a Lévy flight random walk strategy to seed position equations [52], collectively refine TSA's performance and adaptability in optimization tasks.

- Algorithm applications: TSA and its various iterations are applied across a wide array of fields. For instance, CTSA is adept at handling constrained optimization problems by leveraging Deb's rules for tree and seed selection [53]. Meanwhile, DTSA integrates swap, shift, and symmetry transformation operators to tackle permutation-coded optimization problems [54]. In financial risk assessment, Jiang introduces the sinhTSA-MLP model for identifying credit default risks with remarkable precision [55]. Moreover, in the medical domain, Aslan proposes the TSA-ANN structure for precise COVID-19 diagnosis, optimizing artificial neural networks to classify deep architectural features [56].

The broad range of applications demonstrates the adaptability and efficacy of TSA and its derivatives in tackling multifaceted challenges spanning various domains. Continuous improvements and fine-tuning bolster TSA's optimization prowess, solidifying its role as a pivotal tool in diverse problem-solving contexts.

2.3. An Overview of PSO

PSO is a popular metaheuristic algorithm designed for solving global optimization problems. Proposed by Dr. Eberhart and Dr. Kennedy in 1995, PSO draws inspiration from the social behavior of bird flocks and fish schools [57]. The working principle of the PSO algorithm can be briefly summarized as follows.

In the initialization phase, particles are randomly generated within the search space. Each particle is characterized by its position $X_i = (x_{i1}, x_{i2}, \ldots, x_{id}, \ldots, x_{iD})$ and velocity $V_i = (v_{i1}, v_{i2}, \ldots, v_{id}, \ldots, v_{iD})$, where D denotes the problem's dimensionality and i is the index of the particle. Next, the fitness of each particle is evaluated by applying the objective function to its current position by Equation (4).

$$\text{Fitness}_i = \text{Objective Function}(X_i) \tag{4}$$

Subsequently, the local best position (P_{best_i}) for each particle and the global best position (G_{best}) for the entire swarm are updated based on fitness improvement, as shown in Equation (5) below:

$$\begin{aligned} P_{\text{best}_i} &= \begin{cases} X_i, & \text{if Fitness}_i < \text{Fitness}_{\text{best}_i} \\ P_{\text{best}_i}, & \text{otherwise} \end{cases} \\ G_{\text{best}} &= X_{\min(\text{Fitness})} \end{aligned} \tag{5}$$

Then, the particle velocities and positions are updated using the velocity and position update formulas, considering inertia (w), cognitive (c_1), and social (c_2) components, as shown below in Equation (6):

$$V_i = wV_i + c_1 r_1 (P_{\text{best}_i} - X_i) + c_2 r_2 (G_{\text{best}} - X_i) \tag{6}$$

where w is the inertia weight, c_1 and c_2 are learning rates, and r_1 and r_2 are random numbers between 0 and 1. Finally, through iterative optimization, the swarm converges towards the global optimum or an approximate solution.

3. Methods

3.1. PSO-Inspired Seed Generation Mechanism

The original TSA employed a rudimentary seed generation mechanism based on random perturbations of tree positions within the search space. This approach lacked a systematic and adaptive strategy, which could lead to suboptimal exploration and exploitation. To address these limitations, we introduce a novel PSO-inspired seed generation mechanism; in this mechanism, each tree has a certain initial velocity. Furthermore, in the process of generating seeds, the velocity update of each tree is also affected by ST. As a result, two update strategies emerge: the speed update strategy for local development and the speed update strategy for global development, as depicted in Equation (9) and illustrated in Figure 2. These strategies introduce a dynamic and adaptive element to speed generation, as detailed in Equation (8) and demonstrated in Figure 3.

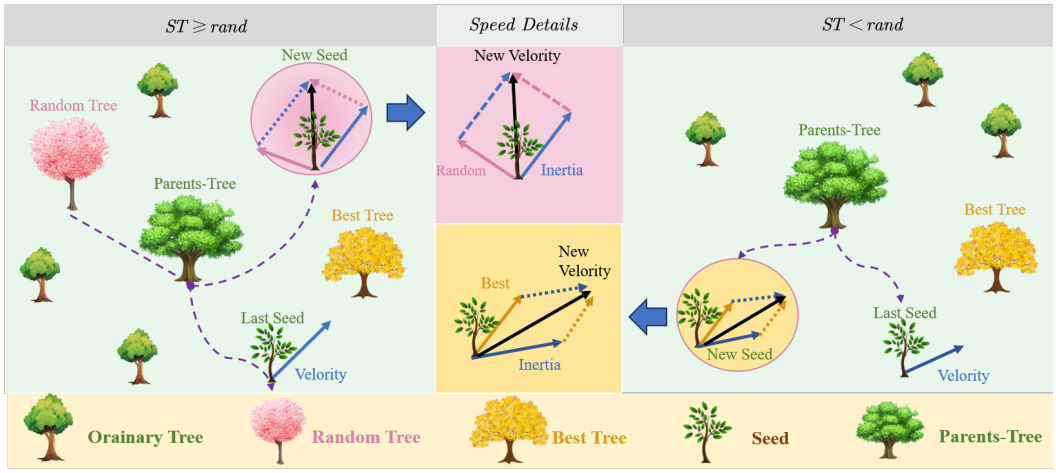


Figure 2. The principle of velocity-driven seed generation.

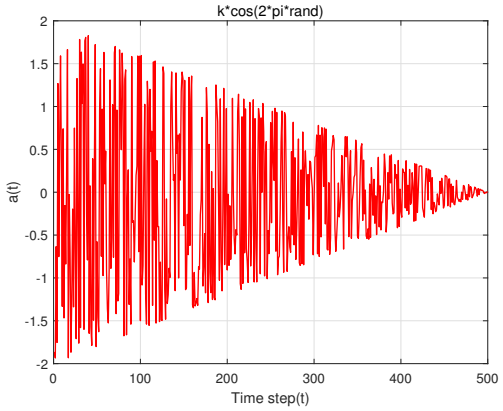


Figure 3. The variation in k * cos (2 * pi * rand) with the number of iterations.

$$v_{j+1,d} = \begin{cases} w \cdot v_{j,d} + \text{rand} \cdot (\text{best\_params}(d) - \text{trees}(i,d)) \cdot k \cdot \cos(2\pi \cdot \text{rand}), & \text{if } ST < 0.1 \\ w \cdot v_{j,d} + \text{rand} \cdot (\text{trees}(r,d) - \text{trees}(i,d)) \cdot k \cdot \cos(2\pi \cdot \text{rand}), & \text{if } ST \geq 0.1 \end{cases} \quad (7)$$

$$k = 2 - 2 \cdot \left(\frac{t}{\text{Max\_Gen}} \right) \quad (8)$$

In the speed vector update mechanism of seeds, $w \cdot v(j,d)$ represents the inertia or influence of the previous velocity on the current velocity. The inertia weight (w) ensures that the algorithm retains some information from the previous iteration, contributing to the smooth transition between consecutive velocities and aiding in the stability of the optimization process. $rand \cdot (trees(r,d) - trees(i,d))$ introduces a random perturbation to

the velocity, facilitating exploration of the solution space. The subtraction $trees(r,d) - trees(i,d)$ signifies the difference in positions between the current tree (i) and a randomly chosen neighboring tree (r). Multiplying by *rand* introduces a random scaling factor, adding stochasticity to the algorithm. $k \cdot \cos(2\pi \cdot \text{rand})$ utilizes k as a control parameter to adjust the amplitude of the cosine term. The cosine function introduces periodicity and oscillatory behavior to the velocity update, aiding exploration by allowing the algorithm to escape local optima and explore diverse regions of the solution space. Multiplying by k provides adaptive control over the amplitude of the cosine term.

Meanwhile, the updated speed will be incorporated into Equations (2) and (3), as described in Equation (9), utilizing velocity vector integration to update the seed position. The introduction of velocity vectors in the optimization process is pivotal for enhancing the exploration–exploitation balance and adaptability of the algorithm by integrating velocity vectors for updating seed positions.

$$\text{seeds}(j,d) = \begin{cases} \text{trees}(i,d) + ((\text{best\_params}(d) - \text{trees}(r,d)) \cdot (\text{rand} - 0.5) \cdot 2) + v(j+1,d), & \text{if } ST < 0.1 \\ \text{trees}(i,d) + ((\text{trees}(r,d) - \text{trees}(i,d)) \cdot (\text{rand} - 0.5) \cdot 2) + v(j+1,d), & \text{if } ST \geq 0.1 \end{cases} \quad (9)$$

3.2. Adaptive Velocity Adaptation Mechanism Based on Count Parameters

The static nature of TSA, with a fixed velocity update strategy, limits its adaptability across various optimization scenarios, leading to issues such as premature convergence and insufficient exploration. The introduced count-based adaptive mechanism dynamically adjusts the algorithm's behavior, effectively balancing exploration and exploitation. A pivotal enhancement is introduced through the incorporation of a counter, denoted as count, to monitor changes in the global optimal tree. If the global optimal tree is replaced, the counter resets to zero. Conversely, if the global optimal tree remains unchanged for a consecutive duration, when the count reaches a threshold of 15, it indicates being trapped in a local optimum. To overcome this challenge and facilitate escape from local optima, the velocity vector is inverted (Equation (10) and Figure 4). This inversion disrupts the pattern leading to local optima, enabling seeds to break free from suboptimal solutions. This innovative approach significantly bolsters the algorithm's global search capability by promoting diversity and exploration, enhancing resilience against convergence to suboptimal solutions.

$$V_{(j+1,d)} = \begin{cases} -V_{(j+1,d)} & Count > 15 \\ V_{(j+1,d)} & Count \leq 15 \end{cases} \quad (10)$$

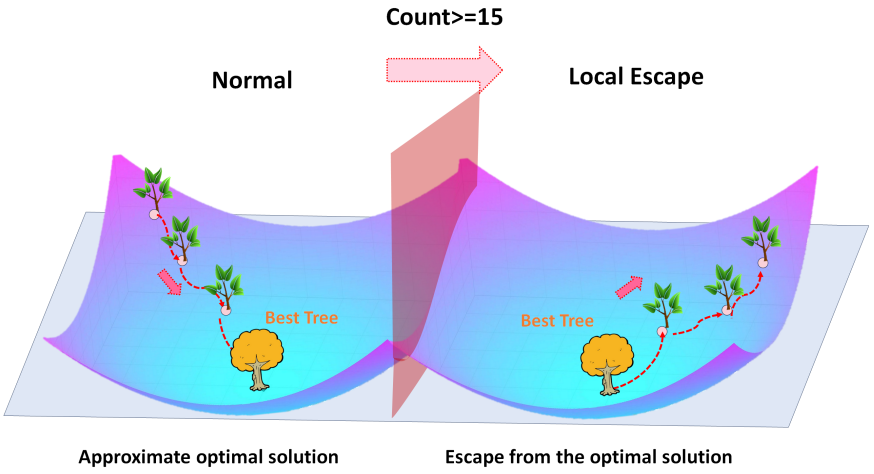


Figure 4. Adaptive velocity adaptation mechanism based on count parameters.

3.3. Population-Based Evolutionary Strategy with Information Exchange Mechanism

Due to the standard TSA's slow convergence speed and diminished population diversity in later iterations, achieving the global optimal value becomes challenging. To address this, arithmetic crossover and natural selection strategies are incorporated to enhance tree diversity, convergence speed, and accuracy illustrated in Figure 5. The trees are arranged by selecting the absolute values of all particle fitness values, as shown in Equation (11), and divided according to Equation (12).

$$[|P_1|, |P_2|, |P_3|, \ldots, |P_N|] \tag{11}$$

$$\begin{aligned} \zeta_1 &= \{|P_1|, |P_2|, |P_3|, \ldots, |P_{\frac{5}{N}}|\} \\ \zeta_2 &= \{|P_{\frac{N}{2}+1}|, \ldots, |P_{\frac{N}{2}}|\} \\ \zeta_3 &= \{|P_{\frac{N}{2}+1}|, \ldots, |P_{\frac{4N}{5}}|\} \\ \zeta_4 &= \{|P_{\frac{4N}{5}+1}|, \ldots, |P_N|\} \end{aligned} \tag{12}$$

In Equation (12), the fitness values are partitioned into four components: ξ_1, ξ_2, ξ_3, and ξ_4. The stochastic nature of the TSA algorithm categorizes fitness values into two groups: favorable (ξ_1, ξ_2) and unfavorable (ξ_3, ξ_4). Specifically, ξ_1 and ξ_4 each constitute 20% of the total population, while ξ_2 and ξ_3 each represent 30%.

The crossover operation is applied to ξ_2 and ξ_3, facilitating the fusion of trees with favorable and unfavorable fitness values. This approach generates offspring with increased diversity and propels less fit trees towards improved fitness values.

Simultaneously, natural selection acts upon ξ_1 and ξ_4. This process involves directly replacing the velocity and position of less fit trees (ξ_4) with those of more fit trees (ξ_1), analogous to reducing the entire population by 20%, significantly accelerating the convergence speed of particles.

The specific strategies are as follows:

- Arithmetic crossover: This paper proposes a novel crossover strategy for trees based on the crossover strategy of Differential Evolution (DE). The update equations for the position $x_{\xi 3}$ and velocity $v_{\xi 3}$ of trees at the locations ξ2 and ξ3, respectively, are defined as follows:

$$x_{\xi 3} = \text{rand} \times x_{\xi 2} + (1 - \text{rand}) \times x_{\xi 3} \tag{13}$$

$$v_{\xi 3} = \text{rand} \times v_{\xi 2} + (1 - \text{rand}) \times v_{\xi 3} \tag{14}$$

  where $x_{\xi 2}$, $x_{\xi 3}$, $v_{\xi 2}$, and $v_{\xi 3}$ represent the positions and velocities of trees, as defined in Equations (11) and (12), and rand is a random number uniformly distributed in the range [0, 1].
- Natural selection: To expedite the convergence speed of trees, a mechanism is employed whereby well-performing trees replace less effective ones. The procedure is expressed as Equations (17) and (18):

$$x_{\xi 4} = x_{\xi 1} \tag{15}$$

$$v_{\xi 4} = v_{\xi 1} \tag{16}$$

  where $x_{\xi 1}$, $x_{\xi 4}$, $v_{\xi 1}$, and $v_{\xi 4}$ represent the positions and velocities of trees as defined in Equation (12). This natural selection process involves the direct replacement of the position and velocity of less effective trees (ξ4) with those of more effective trees (ξ1), significantly accelerating the convergence of trees.

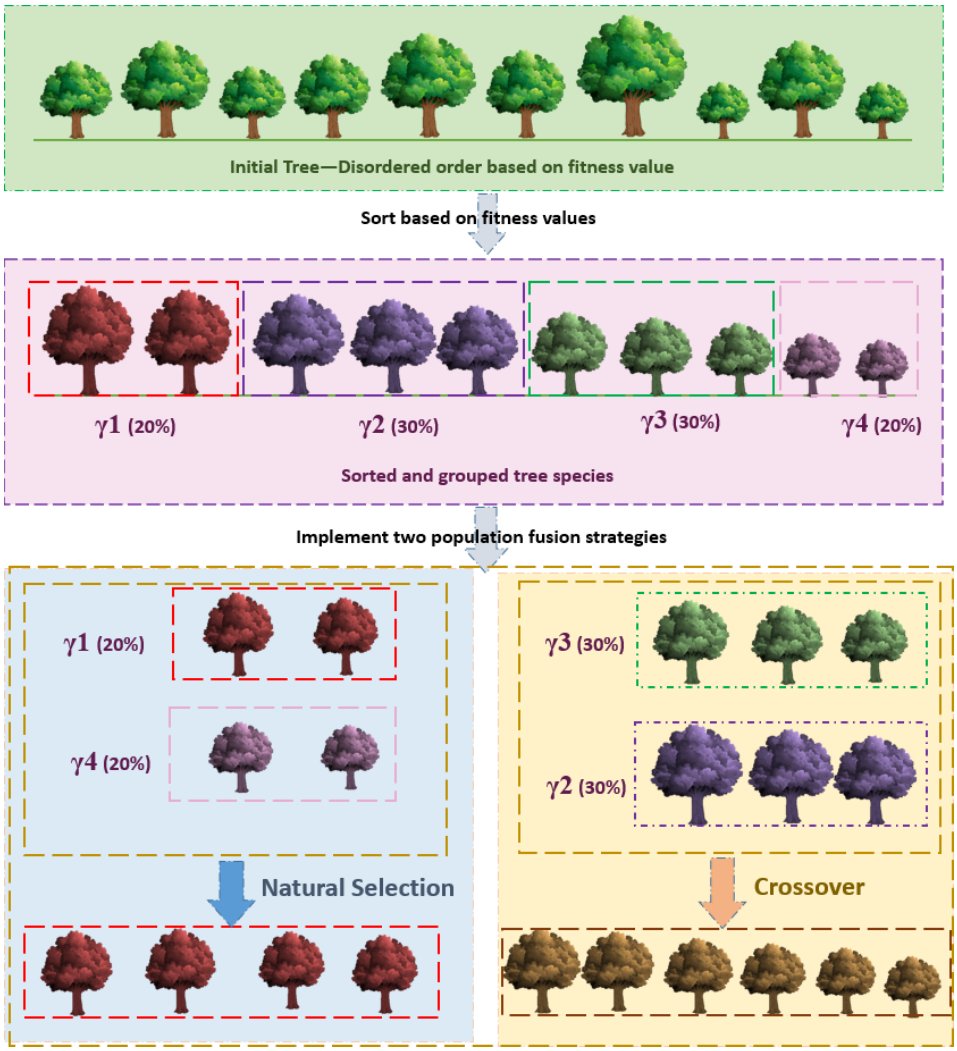


Figure 5. Schematic of population-based evolutionary strategy with information exchange.

3.4. DTSA: A Novel Tree-Seed Algorithm

The proposed algorithm represents a substantial improvement over the original TSA by addressing its limitations through a multifaceted approach. The adaptive velocity-driven seed generation method introduces a PSO-inspired mechanism, integrating velocity vectors into seed generation to dynamically update positions. This enhancement significantly improves exploration and exploitation, providing adaptability and a refined global search capability. The count-based adaptive velocity update method introduces a dynamic adjustment mechanism based on a counter, effectively preventing premature convergence and enhancing exploration by strategically inverting velocity vectors. Lastly, the tree population integrated evolutionary strategy tackles slow convergence and reduced diversity issues by incorporating arithmetic crossover and natural selection. This integrated evolutionary approach transforms the population structure, accelerating convergence and improving overall algorithm performance. Together, these methods contribute to a more dynamic, adaptive, and efficient optimization algorithm, effectively mitigating the drawbacks of the original TSA.

The flowchart illustrating DTSA is presented in Figure 6, while the pseudo-code is provided in Algorithm 1.

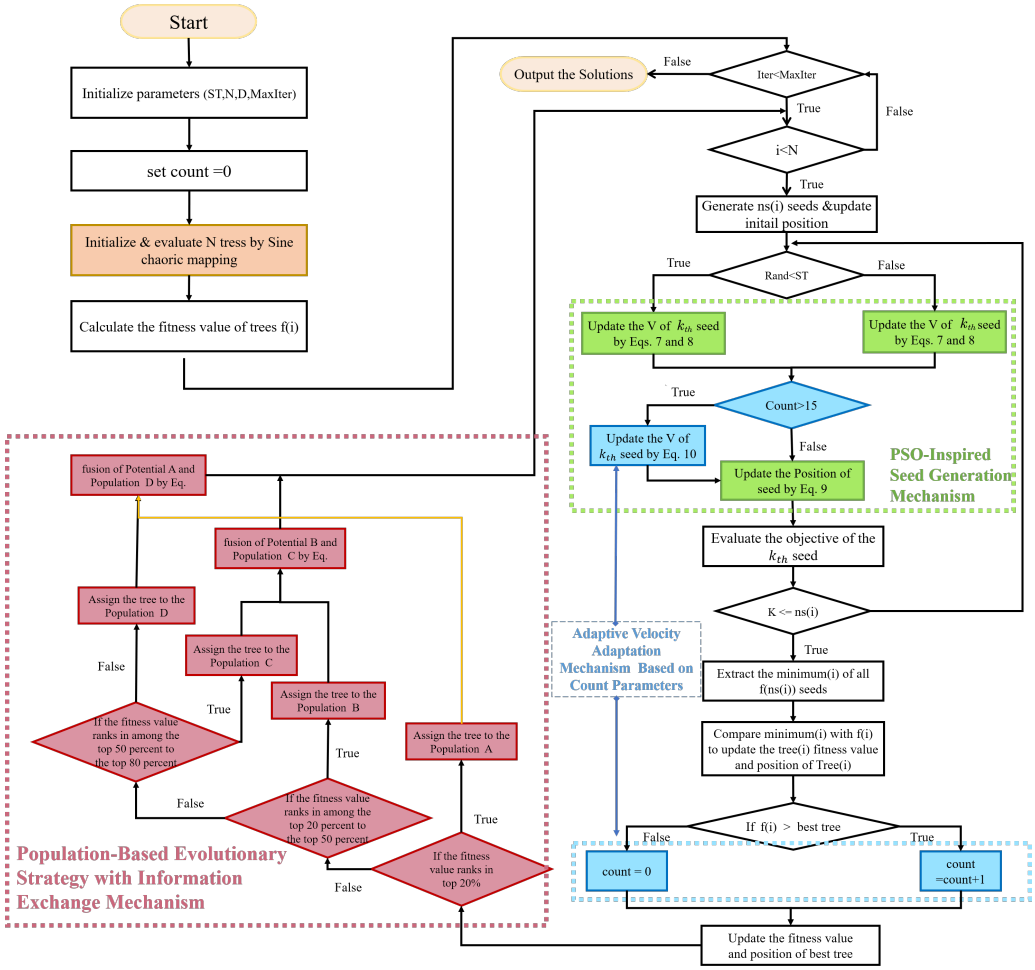


Figure 6. The flowchart for the DTSA.

3.5. Time Complexity Analysis of DTSA

To study the time complexity of DTSA, it is crucial to understand the main components of the code and their frequencies of execution. The DTSA primarily comprises initialization, iterative updates, crossover operations, and natural selection. Here, is an analysis of the time complexities involved in each part.

Initialization: This stage employs Chebyshev chaotic mapping to generate random numbers for each dimension of every tree. With two nested loops—outer iterating N times (number of trees) and inner D times (problem dimensions)—the time complexity is estimated at $O(N \times D)$.

Iterative update phase: Within a loop iterating up to a maximum of $MaxIters$ times, several key steps unfold. First, adaptive weight calculation occurs, which is computationally light with a time complexity of $O(1)$. Proceeding to seed generation, each tree produces a variable number of seeds within a dynamic range defined by 'low' and 'high', resulting in a complexity of $O(N \times (low + high) \times D)$. Estimating NS, the average number of seeds per tree as $\frac{N \times (low+high)}{2}$, calculating the fitness for each seed incurs a cost of $O(NS \times D)$. Updates to tree positions and tracking the best solution both occur with a linear time complexity, $O(N)$, while updating a counter is negligible at $O(1)$. Consequently, the aggregate time complexity for this iterative process is approximately $O(MaxIters \times N \times D)$.

Crossover and natural selection: This phase initiates with sorting trees based on fitness, demanding $O(N \cdot \log(N))$ time. Segregation of the population follows swiftly with $O(N)$ complexity. Crossover operations and natural selection, both executed in $O(N \times D)$ time, precede an analogous double-loop operation updating tree velocities and

positions, also in $O(N \times D)$. This segment, integrating sorting, population manipulation, genetic mechanisms, and status refreshes, accumulates to a total time complexity around $O(MaxIters \times (N \times D))$.

In summary, while DTSA's time complexity mirrors TSA's in most procedural aspects, the crossover and natural selection phases introduce additional computational demands without fundamentally elevating the overall time complexity of the algorithm.

Algorithm 1 The pseudo-code of the DTSA

Step 1. Initialize of the population
Set the initial number of zones and the dimensions of the problem;
Set the ST parameter for the algorithm and termination conditions;
Evaluates the location of the tree against the specified target function;
Creation1:
Generating the positions and velocities of all trees and seeds through Sine chaotic mapping
End Creation1
Step 2. Search with seeds
for all trees **do**
  Decide the number of seeds produced for this tree;
  for all seeds **do**
    for all dimensions **do**
      if rand $<$ ST **then**
        Creation2:
        Generate seed speed through Equations (9) and (10)
        Generate the position of tree by Equation (11)
      else
        Generate seed speed through Equations (9) and (10)
        Generate the position of tree by Equation (11)
        End Creation2
      end if
      Creation3:
      if Count $\geq$ 15 **then**
        The speed of updating seeds by Equation (12)
      end if
      End Creation3
    end for
  end for
  Choose the best seeds and compare them with the trees;
  If the seed is in a better position than the tree, replace the tree with the seed.
end for
Step 3. Select of Best Solution
Select the best solution of the population;
if the new best solution number is lower than the previous best solution number **then**
  Count = Count + 1
else
  Count = 0
  the new best solution will replace the previous best solution
end if
Creation4:
Divide trees into good and bad groups by Equations (11) and (12)
Perform crossover operation between good and bad groups by Equations (13) and (14)
Perform natural selection operation between bad groups by Equations (15) and (16)
End Creation4
Step 4. Test Termination condition
If the condition is not met, return to Step 2.
Step 5. Report
Report the best solution.

4. Results and Discussion

In this section, we present a series of experiments to assess the performance of the proposed DTSA. We analyze the advantages of DTSA based on six comprehensive experimental results. Additionally, Section 4.1 outlines the parameter configurations utilized in the upcoming experiments. The subsequent experiments' parameter settings are introduced

in detail, providing a comprehensive overview of the experimental setup. Subsequently, in Section 4.2, we initiate the exploration with the qualitative analysis of DTSA. Subsequently, in Section 4.3, we delve into the quantitative aspects of the analysis. Section 4.4 conducts an in-depth examination of the experimental data from the preceding quantitative analysis. Further reinforcing the significance of DTSA, Section 4.5 presents statistical experiments, demonstrating its superior performance compared to other algorithms. In Section 4.6, we extend the validation of DTSA by applying it to address three real-world problems.

4.1. Experiment Setting

DTSA underwent a comprehensive performance assessment by being pitted against various cutting-edge algorithms and several modifications. Notable contenders included EST-TSA [50], fb-TSA [51], TSA [34], STSA [42], MTSA [38], GA [58], PSO [57], GWO [59], BA [60], BOA, and RSA [61].

To ensure experimental fairness, this paper meticulously records the datasets and parameters employed by the comparison algorithms in the original research, as presented in Table 1. The experimental dataset is based on the IEEE CEC 2014 benchmark function, as indicated by the statistical results, where $F1$–$F3$ are unimodal functions, $F4$–$F16$ are simple multimodal functions, $F17$–$F23$ are mixed functions, and $F24$–$F30$ are combined functions [62]. Further details regarding the CEC 2014 benchmark function can be found in Tables 2 and 3.

Table 1. The initial parameters of comparative algorithms.

Algorithm	Parameter	Value
DTSA	ST	0.1
EST-TSA	ST	0.1
fb-TSA	ST	0.1
TSA	ST	0.1
STSA	ST	0.1
MTSA	ST	0.1
GWO	a	Linearly decreased from 2 to 0
GA	Type Selection Crossover Mutation	Real coded Roulette wheel $Probability = 0.7$ $Probability = 0.2$
BOA	p	0.8
RSA	Evolutionary Sense Sensitive parameter controlling the exploration accuracy Sensitive parameter controlling the exploitation accuracy	$2 \times randn \times (1 - (iter/maxiter))$ 0.05 0.1
HHO	E_0 β	Range from [−1,1] 1.5
GOA	α p Power exponent Sensory modality	Linearly decreased from 2 to 0 0.1 0.01
DBO	Producers	0.2
WFO	Probability of laminar flow Probability of spiral flow in turbulent flow	0.3 0.7
SO	Threshold Threshold2 C_1 C_2 C_3	0.25 0.6 0.5 0.05 2

Table 2. Definitions of the basic IEEE benchmark functions.

Function Name	Function Details
High Conditioned Elliptic Function	$f_1(X) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} X_i^2$
Bent Cigar Function	$f_2(X) = X_1^2 + 10^6 \sum_{i=2}^{D} X_i^2$
Discus Function	$f_3(X) = 10^6 X_1^2 + \sum_{i=2}^{D} X_i^2$
Rosenbrock's Function	$f_4(X) = \sum_{i=1}^{D-1} (100(X_i^2 - X_{i+1})^2 + (X_i - 1)^2)$
Ackley's Function	$f_5(X) = -20 \exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} X_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi X_i)\right) + 20 + e$
Weierstrass Function	$f_6(X) = \sum_{i=1}^{D} \left(\sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k (X_i + 0.5))]\right) - D \sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k \times 0.5)]; \quad a = 0.5, b = 3, k_{\max} = 20$
Griewank's Function	$f_7(X) = \sum_{i=1}^{D} \frac{X_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{X_i}{\sqrt{i}}\right) + 1$
Rastrigin's Function	$f_8(X) = \sum_{i=1}^{D} (X_i^2 - 10\cos(2\pi X_i) + 10)$
Modified Schwefel's Function	$f_9 = 418.9829 \times D - \sum_{i=1}^{D} g(z_i); z_i = x_i + 4.209687462275036 \times 10^{+002}$
Katsuura Function	$f_{10}(X) = \frac{10}{D^2} \prod_{i=1}^{D} \left(1 + i \sum_{j=1}^{32} \frac{\lvert 2^j X_i - \text{round}(2^j X_i) \rvert}{2^j}\right)^{\frac{10}{D^{1.2}}} - \frac{10}{D^2}$
HappyCat Function	$f_{11}(X) = \left\lvert \sum_{i=1}^{D} X_i^2 - D \right\rvert^{1/4} + (0.5\sum_{i=1}^{D} X_i^2 + \sum_{i=1}^{D} X_i)/D + 0.5$
HGBat Function	$f_{12}(X) = \left\lvert \left(\sum_{i=1}^{D} X_i^2\right)^2 - \left(\sum_{i=1}^{D} X_i\right)^2 \right\rvert^{1/2} + (0.5\sum_{i=1}^{D} X_i^2 + \sum_{i=1}^{D} X_i)/D + 0.5$
Expanded Griewank's plus Rosenbrock's Function	$f_{13} = f_7(f_4(X_1, X_2)) + f_7(f_4(X_2, X_3)) + \ldots + f_7(f_4(X_{D-1}, X_D)) + f_7(f_4(X_D, X_1))$
Expanded Scaffer's F6 Function	$g(X, Y) = 0.5 + \frac{\sin^2(\sqrt{X^2+Y^2}) - 0.5}{(1+0.001(X^2+Y^2))^2}$; $f_{14} = g(X_1, X_2) + g(X_2, X_3) + \ldots + g(X_{D-1}, X_D) + g(X_D, X_1)$

Table 3. Benchmark functions of IEEE CEC 2014.

A. Unimodal Functions	
Rotated High Conditioned Elliptic Function	$F_1(x) = f_1(M(x - o_1)) + 100$
Rotated Bent Cigar Function	$F_2(x) = f_2(M(x - o_2)) + 200$
Rotated Discus Function	$F_3(x) = f_3(M(x - o_3)) + 300$
B. Multimodal Functions	
Shifted and Rotated Rosenbrock's Function	$F_4(x) = f_4(M(\frac{2.048(x - o_4)}{100}) + 1) + 400$
Shifted and Rotated Ackley's Function	$F_5(x) = f_5(M(x - o_5)) + 800$
Shifted and Rotated Weierstrass Function	$F_6(x) = f_6(M(\frac{0.5(x - o_6)}{100})) + 600$
Shifted and Rotated Griewank's Function	$F_7(x) = f_7(M(\frac{600(x - o_7)}{100})) + 700$
Shifted Rastrigin's Function	$F_8(x) = f_8(M(\frac{5.12(x - o_8)}{100})) + 700$
Shifted and Rotated Rastrigin's Function	$F_9(x) = f_8(M(\frac{5.12(x - o_9)}{100})) + 900$
Shifted Schwefel's Function	$F_{10}(x) = f_9(M(\frac{1000(x - o_{10})}{100})) + 1000$
Shifted and Rotated Schwefel's Function	$F_{11}(x) = f_9(M(\frac{1000(x - o_{11})}{100})) + 1100$
Shifted and Rotated Katsuura Function	$F_{12}(x) = f_{10}(M(\frac{5(x - o_{12})}{100})) + 1200$
Shifted and Rotated HappyCat Function	$F_{13}(x) = f_{11}(M(\frac{5(x - o_{13})}{100})) + 1300$
Shifted and Rotated HGBat Function	$F_{14}(x) = f_{12}(M(\frac{5(x - o_{14})}{100})) + 1400$
Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	$F_{15}(x) = f_{13}(M(\frac{5(x - o_{15})}{100}) + 1) + 1500$
Shifted and Rotated Expanded Scaffer's F6 Function	$F_{16}(x) = f_{14}(M((x - o_{16})) + 1) + 1600$

Table 3. *Cont.*

C. Hybrid Functions	
$F_{17} = f_9(M_1Z_1) + f_8(M_2Z_2) + f_1(M_3Z_3) + 1700$	p = [0.3,0.3,0.4]
$F_{18} = f_2(M_1Z_1) + f_12(M_2Z_2) + f_8(M_3Z_3) + 1800$	p = [0.3,0.3,0.4]
$F_{19} = f_7(M_1Z_1) + f_6(M_2Z_2) + f_4(M_3Z_3) + f_14(M_4Z_4) + 1900$	p = [0.2,0.2,0.3,0.3]
$F_{20} = f_{12}(M_1Z_1) + f_3(M_2Z_2) + f_{13}(M_3Z_3) + f_8(M_4Z_4) + 2000$	p = [0.2,0.2,0.3,0.3]
$F_{21} = f_{14}(M_1Z_1) + f_{12}(M_2Z_2) + f_4(M_3Z_3) + f_9(M_4Z_4) + f_1(M_5Z_5) + 2100$	p = [0.1,0.2,0.2,0.2,0.3]
$F_{22} = f_{10}(M_1Z_1) + f_{11}(M_2Z_2) + f_{13}(M_3Z_3) + f_9(M_4Z_4) + f_5(M_5Z_5) + 2200$	p = [0.1,0.2,0.2,0.2,0.3]
D. Composition Functions	
$F_{23} = w_1 * F'_4(x) + w_2 * [1e^{-6}F'_1(x) + 100] + w_3 * [1e^{-26}F'_2(x) + 200]$ $+w_4 * [1e^{-6}F'_3(x) + 300] + w_5 * [1e^{-6}F'_1(x) + 400] + 2300$	σ = [10,20,30,40,50]
$F_{24} = w_1 * F'_{10}(x) + w_2 * [F'_9(x) + 100] + w_3 * [F'_{14}(x) + 200] + 2400$	σ = [20,20,20]
$F_{25} = w_1 * 0.25F'_{11}(x) + w_2 * [F'_9(x) + 100] + w_3 * [1e^{-7}F'_1(x) + 200] + 2500$	σ = [10,30,50]
$F_{26} = w_1 * 0.25F'_{11}(x) + w_2 * [F'_{13}(x) + 100] + w_3 * [1e^{-7}F'_1(x) + 200]$ $+w_4 * [2.5F'_6(x) + 300] + w_5 * [10F'_7(x) + 400] + 2600$	σ = [10,10,10,10,10]
$F_{27} = w_1 * 10F'_{14}(x) + w_2 * [10F'_9(x) + 100] + w_3 * [2.5F'_{11}(x) + 200]$ $+w_4 * [25F'_6(x) + 300] + w_5 * [1e^{-6}F'_1(x) + 400] + 2700$	σ = [10,10,10,20,20]
$F_{28} = w_1 * 2.5F'_{15}(x) + w_2 * [10F'_{13}(x) + 100] + w_3 * [2.5F'_{11}(x) + 200]$ $+w_4 * [5e^{-4}F'_{16}(x) + 300] + w_5 * [1e^{-6}F'_1(x) + 400] + 2800$	σ = [10,20,30,40,50]
$F_{29} = w_1 * F'_{17}(x) + w_2 * [F'_{18}(x) + 100] + w_3 * [F'_{19}(x) + 200] + 2900$	σ = [10,30,50]
$F_{30} = w_1 * F'_{20}(x) + w_2 * [F'_{21}(x) + 100] + w_3 * [F'_{22}(x) + 200] + 3000$	σ = [10,30,50]

4.2. Qualitative Analysis

In order to qualitatively analyze the proposed algorithm, we conducted convergence behavior analysis, population diversity analysis, and exploration and exploitation analysis experiments to observe its performance. At the same time, we selected classical unimodal function $F1$ to evaluate the development ability of the algorithm, and classical multimodal function $F8$ to evaluate the exploration ability of the algorithm, as shown in Figures 7 and 8.

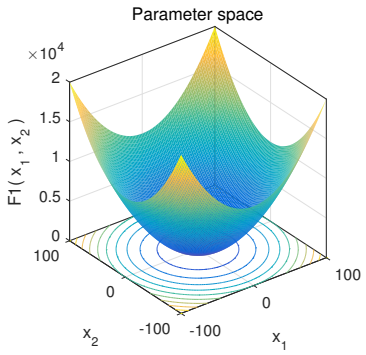


Figure 7. Unimodal function F1.

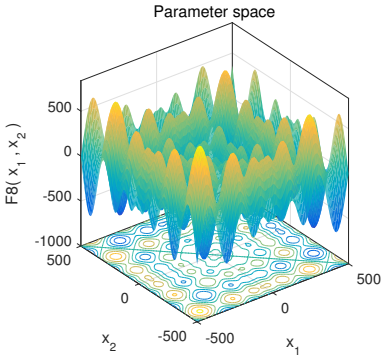
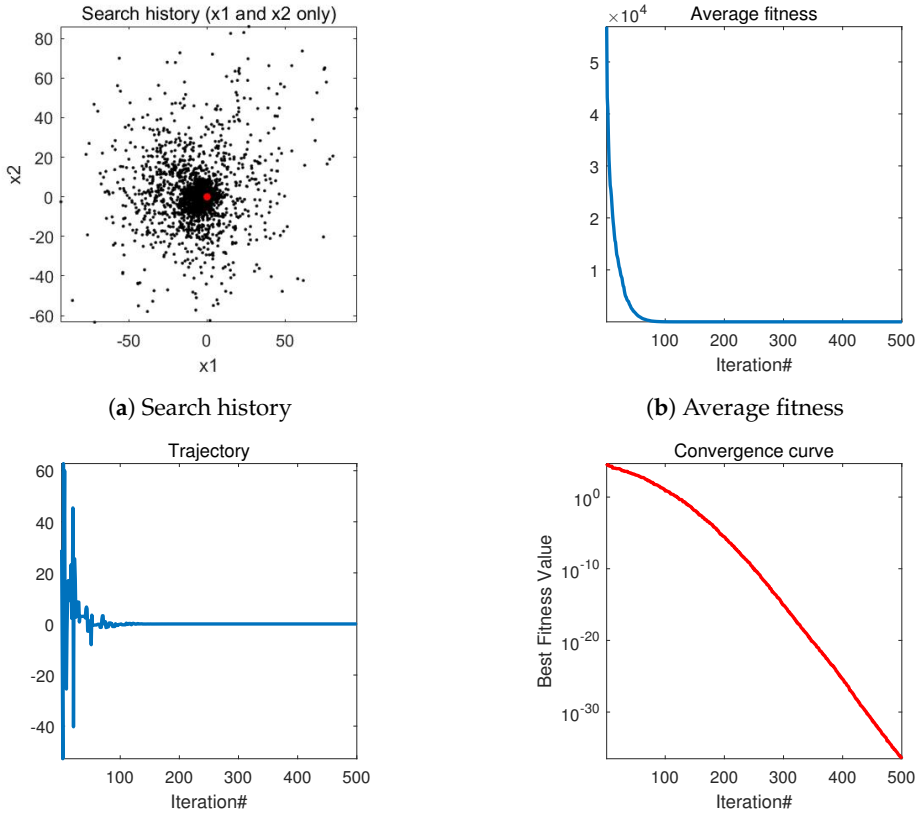


Figure 8. Multimodal function F8.

4.2.1. Convergence Behavior Analysis

In this experiment, there are a total of four convergence analysis graphs (in Figures 9 and 10), the specific meanings of the four pictures of the different functions are as follows:

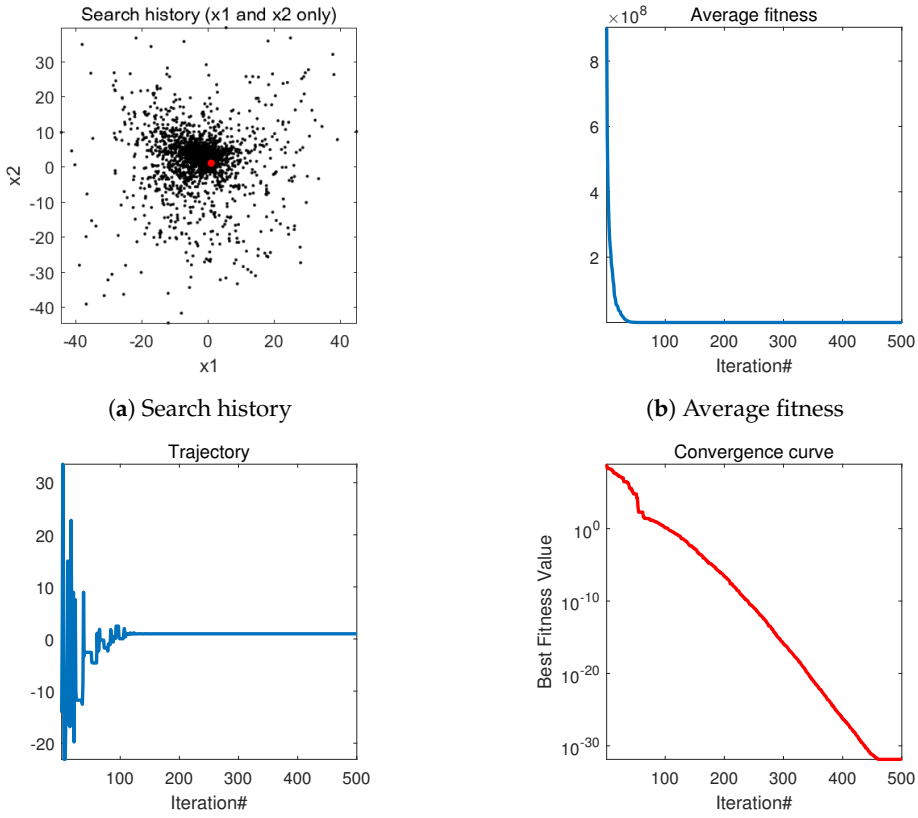
- The first image illustrates the optimization process of the DTSA algorithm. The black dots represent the areas covered by current seeds, while the red dot indicates the best position found, representing the optimal solution. The clustering of black dots around the red dot demonstrates the step-by-step optimization of DTSA towards convergence.
- The second figure depicts the convergence of DTSA, showcasing its rapid convergence towards the optimal solution. The sharp decline in the convergence curve underscores DTSA's efficiency in finding optimal solutions.
- The third graph monitors changes in the first dimension, offering insights into the algorithm's behavior and its avoidance of premature convergence to local optima. Empirical evidence suggests that the DTSA algorithm effectively navigates away from local optima.
- In the fourth graph, the convergence of the mean over multiple iterations is presented. The noticeable decline in the curve indicates the significant overall convergence effect of DTSA, further affirming its efficacy in optimization tasks.



(**a**) Search history (**b**) Average fitness

(**c**) Trajectory (**d**) Convergence curve

Figure 9. The qualitative analysis of DTSA in unimodal function (F1) (search history (**a**), average fitness history (**b**), trajectory history (**c**), convergence curve (**d**)).



(**a**) Search history (**b**) Average fitness

(**c**) Trajectory (**d**) Convergence curve

Figure 10. The qualitative analysis of DTSA in multimodal function (F8) (search history (**a**), average fitness history (**b**), trajectory history (**c**), convergence curve (**d**)).

4.2.2. Population Diversity Analysis

In optimization algorithms, population diversity serves as a pivotal metric for gauging efficiency. Its level has a direct impact on the exploration depth within the algorithm's search space. A heightened population diversity is instrumental in averting entrapment in local optima, preserving the algorithm's prowess for global search. This fosters a more extensive exploration of the search space during initial stages, thereby enhancing the likelihood of discovering the global optimal solution. This broadened search capability enables DTSA to adeptly handle problems with varying shapes and structures, mitigating over-optimization for specific instances. Consequently, the maintenance of appropriate population diversity emerges as a critical factor for ensuring the robustness and global search performance of algorithms. In practical applications, the mathematical formula for measuring population diversity can be used in various ways. In this paper, the dispersion of individual and collective centroids is used to represent population diversity, where Equations (17) and (18) are used to calculate the population diversity I_c.

$$I_c(t) = \sqrt{\sum_{i=1}^{N}\sum_{d=1}^{D}(x_{id}(t) - c_d)^2(t)} \tag{17}$$

$$c_d(t) = \frac{1}{N}\sum_{d=1}^{D}(x_{id}(t) \tag{18}$$

We easily observe from Figures 11 and 12 that in the single-modal function $F1$, both DTSA and TSA exhibit a rapid decrease in population diversity. The algorithms exhibit strong local exploitation capabilities, facilitating the rapid identification of global optimum solutions. Despite this, DTSA demonstrates faster convergence compared to TSA while

preserving higher population diversity throughout the process. For the *F*8 multimodal function, DTSA benefits from the sine chaotic map for population initialization, resulting in increased diversity early in the algorithm and maintaining it through mid-term iterations. Although population diversity diminishes after encountering local optima, the count-based adaptive velocity update strategy enables DTSA to bolster diversity, aiding its escape from such solutions.

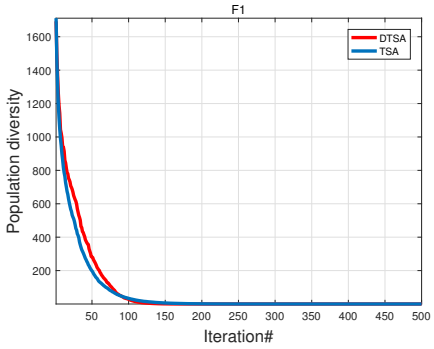


Figure 11. Unimodal functions.

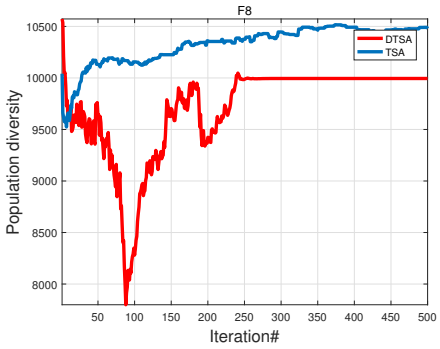


Figure 12. Multimodal functions.

4.2.3. Exploration and Exploitation Analysis

In swarm intelligence optimization algorithms, exploring development analysis is often used to evaluate the exploration capability of the algorithm. The diversity of the swarm across iterations is a key factor in understanding how well the algorithm explores the solution space. The diversity at iteration t (Div_t) can be calculated using the average distance between particles in the swarm by Equation (19):

$$Div_t = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j \neq i}^{N} \sqrt{\sum_{d=1}^{D} (x_{id}^t - x_{jd}^t)^2} \tag{19}$$

This measures the average Euclidean distance between all pairs of particles in the swarm. The exploring development and the exploitation analysis can be defined as the ratio of the diversity at each iteration to the maximum diversity, as shown by Equations (20) and (21):

$$\text{Exploration}(\%) = \frac{Div_t}{Div_{\max}} \tag{20}$$

$$\text{Exploitation}(\%) = \frac{|Div_t - Div_{\max}|}{Div_{\max}} \tag{21}$$

In Figures 13 and 14, a higher value of $Exploration$ indicates better exploration capacity and a higher value of $Exploitation$ indicates better exploitation capacity at iteration t. In comparing DTSA with the original algorithm, both demonstrate rapid development in

single-mode function *F*1, indicating strong development capabilities. For multimode function *F*8, DTSA achieves a balance between development and exploration at 212 iterations, gradually decreasing exploration while maintaining excellent global exploration ability. In contrast, TSA reaches a balance at 1042 iterations, indicating good global exploration but with weaker practical applicability due to a maximum iteration limit of 500 iterations.

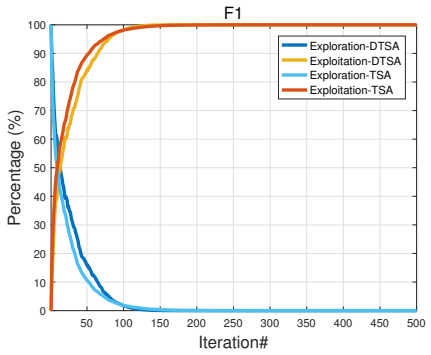


Figure 13. Unimodal functions.

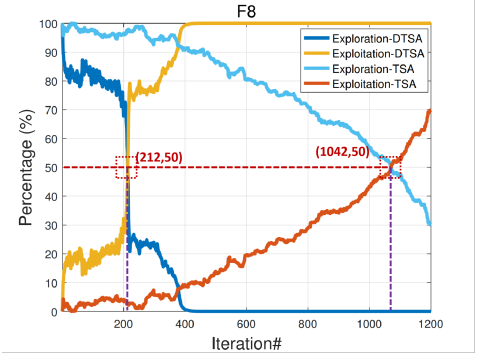


Figure 14. Multimodal functions.

4.3. Quantitative Analysis

In this section, three strategically crafted experiments establish DTSA's superior performance. Section 4.3.1 rigorously compares DTSA with various TSA variants, unequivocally demonstrating its superiority. Subsection 4.3.2 extends the comparison to include additional metaheuristic algorithms, highlighting DTSA's exceptional ability to tackle intricate problems. Subsection 4.3.3 presents boxplots derived from 30 rounds of experiment results, offering visual insights into DTSA's stability across multiple iterations.

4.3.1. Comparative Experiment 1: DTSA versus EST-TSA, fb-TSA, TSA, STSA, and MTSA

The first comparative experiment evaluates basic TSA [34] and its recent variants (EST-TSA [50], fb-TSA [51], STSA [42], and MTSA [38]) to showcase their strengths. Comparative assessments of DTSA and other algorithms across 30, 50, and 100 dimensions are presented in Tables 4–6. Additionally, graphical representations of the convergence curves for these algorithms are depicted in Figures 15–17.

The tables present the average optimal values obtained from 30 experimental iterations, each comprising 500 iterations, offering insights into the convergence performance of the algorithms. This aggregated data are crucial for discerning algorithmic superiority.

Moreover, meticulously documented convergence processes over 500 iterations for each experimental iteration shed light on performance dynamics. By averaging 30 local optimal solutions at each iteration point, the computed average convergence process reveals nuanced performance details. The slopes of these curves provide quantitative assessments of convergence speed.

The results underscore the efficacy of leveraging the fusion mechanism of tree population to enhance population diversity, thereby avoiding local optima. Additionally, refinements in the position update formula (Equation (9)) significantly improve convergence speed compared to previously proposed TSA variants.

Based on the empirical evidence, we confidently assert the superiority of the algorithm introduced in this study over other TSA algorithms and their variants in optimization endeavors.

Table 4. The mean values for the DTSA, EAT-TSA, fb-TSA, TSA, STSA, and MTSA in 30 dimensions.

Function	DTSA	EST-TSA	MTSA	STSA	TSA	fb-TSA
F1	**1.8534** $\times$ **10**$^{+6}$	6.6216 $\times$ 10^{+7}	5.6627 $\times$ 10^{+6}	5.2357 $\times$ 10^{+8}	1.0857 $\times$ 10^{+8}	9.2412 $\times$ 10^{+6}
F2	5.3718 $\times$ 10^{+3}	9.5866 $\times$ 10^{+5}	1.9370 $\times$ 10^{+5}	2.8175 $\times$ 10^{+10}	3.1216 $\times$ 10^{+8}	**2.9438** $\times$ **10**$^{+3}$
F3	**5.3235** $\times$ **10**$^{+2}$	3.3472 $\times$ 10^{+4}	3.6163 $\times$ 10^{+3}	1.0183 $\times$ 10^{+5}	3.9619 $\times$ 10^{+4}	1.9082 $\times$ 10^{+4}
F4	**4.4918** $\times$ **10**$^{+2}$	6.2177 $\times$ 10^{+2}	4.8830 $\times$ 10^{+2}	2.8145 $\times$ 10^{+3}	6.9366 $\times$ 10^{+2}	4.4972 $\times$ 10^{+2}
F5	**5.2003** $\times$ **10**$^{+2}$	5.2101 $\times$ 10^{+2}	5.2098 $\times$ 10^{+2}	5.2096 $\times$ 10^{+2}	5.2102 $\times$ 10^{+2}	5.2100 $\times$ 10^{+2}
F6	6.0130 $\times$ 10^{+2}	6.2733 $\times$ 10^{+2}	6.0111 $\times$ 10^{+2}	6.3899 $\times$ 10^{+2}	6.2873 $\times$ 10^{+2}	**6.0093** $\times$ **10**$^{+2}$
F7	**7.0000** $\times$ **10**$^{+2}$	7.0051 $\times$ 10^{+2}	7.0031 $\times$ 10^{+2}	9.5051 $\times$ 10^{+2}	7.0427 $\times$ 10^{+2}	7.0001 $\times$ 10^{+2}
F8	**8.2189** $\times$ **10**$^{+2}$	9.9125 $\times$ 10^{+2}	8.2342 $\times$ 10^{+2}	1.0774 $\times$ 10^{+3}	9.8800 $\times$ 10^{+2}	8.2388 $\times$ 10^{+2}
F9	**9.2388** $\times$ **10**$^{+2}$	1.1516 $\times$ 10^{+3}	1.0439 $\times$ 10^{+3}	1.2170 $\times$ 10^{+3}	1.1215 $\times$ 10^{+3}	9.7803 $\times$ 10^{+2}
F10	**1.3552** $\times$ **10**$^{+3}$	5.5110 $\times$ 10^{+3}	1.7501 $\times$ 10^{+3}	7.6039 $\times$ 10^{+3}	6.7713 $\times$ 10^{+3}	2.7224 $\times$ 10^{+3}
F11	**3.6458** $\times$ **10**$^{+3}$	7.5516 $\times$ 10^{+3}	8.0938 $\times$ 10^{+3}	8.4115 $\times$ 10^{+3}	8.3971 $\times$ 10^{+3}	6.5521 $\times$ 10^{+3}
F12	1.2028 $\times$ 10^{+3}	**1.2023** $\times$ **10**$^{+3}$	1.2025 $\times$ 10^{+3}	1.2030 $\times$ 10^{+3}	1.2029 $\times$ 10^{+3}	1.2033 $\times$ 10^{+3}
F13	**1.3003** $\times$ **10**$^{+3}$	1.3005 $\times$ 10^{+3}	1.3004 $\times$ 10^{+3}	1.3043 $\times$ 10^{+3}	1.3006 $\times$ 10^{+3}	1.3005 $\times$ 10^{+3}
F14	**1.4003** $\times$ **10**$^{+3}$	1.4003 $\times$ 10^{+3}	1.4003 $\times$ 10^{+3}	1.4800 $\times$ 10^{+3}	1.4004 $\times$ 10^{+3}	1.4003 $\times$ 10^{+3}
F15	**1.5034** $\times$ **10**$^{+3}$	1.5219 $\times$ 10^{+3}	1.5172 $\times$ 10^{+3}	1.7233 $\times$ 10^{+4}	1.5277 $\times$ 10^{+3}	1.5139 $\times$ 10^{+3}
F16	**1.6102** $\times$ **10**$^{+3}$	1.6128 $\times$ 10^{+3}	1.6122 $\times$ 10^{+3}	1.6133 $\times$ 10^{+3}	1.6128 $\times$ 10^{+3}	1.6122 $\times$ 10^{+3}
F17	**2.3889** $\times$ **10**$^{+5}$	1.3751 $\times$ 10^{+6}	5.7574 $\times$ 10^{+5}	1.8705 $\times$ 10^{+7}	1.8190 $\times$ 10^{+6}	4.2697 $\times$ 10^{+5}
F18	**1.8997** $\times$ **10**$^{+3}$	2.1487 $\times$ 10^{+3}	4.5423 $\times$ 10^{+3}	1.2407 $\times$ 10^{+8}	2.1233 $\times$ 10^{+3}	1.9254 $\times$ 10^{+3}
F19	**1.9035** $\times$ **10**$^{+3}$	1.9081 $\times$ 10^{+3}	1.9355 $\times$ 10^{+3}	2.0009 $\times$ 10^{+3}	1.9127 $\times$ 10^{+3}	1.9073 $\times$ 10^{+3}
F20	**4.6994** $\times$ **10**$^{+3}$	2.2861 $\times$ 10^{+4}	7.6132 $\times$ 10^{+3}	4.6643 $\times$ 10^{+4}	1.7044 $\times$ 10^{+4}	1.2748 $\times$ 10^{+4}
F21	**1.2688** $\times$ **10**$^{+5}$	4.7358 $\times$ 10^{+5}	1.5364 $\times$ 10^{+5}	3.0916 $\times$ 10^{+6}	4.3843 $\times$ 10^{+5}	2.0241 $\times$ 10^{+5}
F22	2.4970 $\times$ 10^{+3}	2.7481 $\times$ 10^{+3}	**2.3768** $\times$ **10**$^{+3}$	3.2253 $\times$ 10^{+3}	2.7129 $\times$ 10^{+3}	2.4806 $\times$ 10^{+3}
F23	**2.6152** $\times$ **10**$^{+3}$	2.6155 $\times$ 10^{+3}	2.6152 $\times$ 10^{+3}	2.7273 $\times$ 10^{+3}	2.6191 $\times$ 10^{+3}	2.6152 $\times$ 10^{+3}
F24	2.6244 $\times$ 10^{+3}	**2.6000** $\times$ **10**$^{+3}$	2.6252 $\times$ 10^{+3}	2.6007 $\times$ 10^{+3}	2.6543 $\times$ 10^{+3}	2.6132 $\times$ 10^{+3}
F25	2.7085 $\times$ 10^{+3}	**2.7000** $\times$ **10**$^{+3}$	2.7066 $\times$ 10^{+3}	2.7518 $\times$ 10^{+3}	2.7238 $\times$ 10^{+3}	2.7089 $\times$ 10^{+3}
F26	**2.7003** $\times$ **10**$^{+3}$	2.7527 $\times$ 10^{+3}	2.7004 $\times$ 10^{+3}	2.7040 $\times$ 10^{+3}	2.7007 $\times$ 10^{+3}	2.7004 $\times$ 10^{+3}
F27	3.0204 $\times$ 10^{+3}	3.3265 $\times$ 10^{+3}	3.1028 $\times$ 10^{+3}	3.9204 $\times$ 10^{+3}	3.2294 $\times$ 10^{+3}	**3.0030** $\times$ **10**$^{+3}$
F28	**3.6667** $\times$ **10**$^{+3}$	4.0400 $\times$ 10^{+3}	3.6696 $\times$ 10^{+3}	5.5382 $\times$ 10^{+3}	4.0981 $\times$ 10^{+3}	3.7407 $\times$ 10^{+3}
F29	**3.9272** $\times$ **10**$^{+3}$	7.2542 $\times$ 10^{+4}	4.0237 $\times$ 10^{+3}	2.4545 $\times$ 10^{+7}	1.7331 $\times$ 10^{+5}	4.0245 $\times$ 10^{+3}
F30	**4.5599** $\times$ **10**$^{+3}$	2.3030 $\times$ 10^{+4}	4.9883 $\times$ 10^{+3}	4.8079 $\times$ 10^{+5}	2.1804 $\times$ 10^{+4}	4.7449 $\times$ 10^{+3}
Rank first	23	3	1	0	0	3

Table 5. The mean values for the DTSA, EAT-TSA, fb-TSA, TSA, STSA, and MTSA in 50 dimensions.

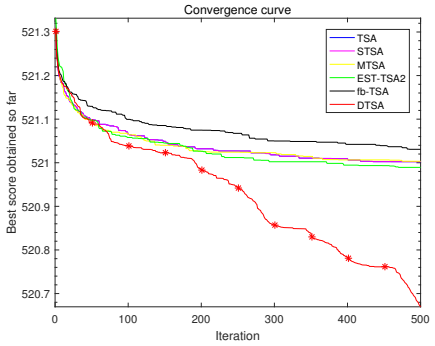
Function	DTSA	EST-TSA	MTSA	STSA	TSA	fb-TSA
F1	**3.9446** $\times$ **10**$^{+6}$	3.9755 $\times$ 10^{+8}	1.4848 $\times$ 10^{+7}	2.3766 $\times$ 10^{+9}	4.3731 $\times$ 10^{+8}	1.8934 $\times$ 10^{+7}
F2	**4.0065** $\times$ **10**$^{+3}$	9.7082 $\times$ 10^{+8}	7.8296 $\times$ 10^{+6}	1.1836 $\times$ 10^{+11}	8.9880 $\times$ 10^{+9}	4.8431 $\times$ 10^{+4}
F3	**7.4047** $\times$ **10**$^{+3}$	1.1842 $\times$ 10^{+5}	6.6952 $\times$ 10^{+4}	2.7862 $\times$ 10^{+5}	1.1875 $\times$ 10^{+5}	9.0153 $\times$ 10^{+4}
F4	**5.2444** $\times$ **10**$^{+2}$	1.4599 $\times$ 10^{+3}	5.2644 $\times$ 10^{+2}	3.4377 $\times$ 10^{+4}	1.9098 $\times$ 10^{+3}	5.2544 $\times$ 10^{+2}
F5	**5.2024** $\times$ **10**$^{+2}$	5.2117 $\times$ 10^{+2}	5.2121 $\times$ 10^{+2}	5.2118 $\times$ 10^{+2}	5.2121 $\times$ 10^{+2}	5.2121 $\times$ 10^{+2}
F6	**6.0826** $\times$ **10**$^{+2}$	6.5543 $\times$ 10^{+2}	6.1059 $\times$ 10^{+2}	6.7526 $\times$ 10^{+2}	6.5902 $\times$ 10^{+2}	6.2356 $\times$ 10^{+2}
F7	**7.0000** $\times$ **10**$^{+2}$	7.0925 $\times$ 10^{+2}	7.0108 $\times$ 10^{+2}	1.8056 $\times$ 10^{+3}	7.8536 $\times$ 10^{+2}	7.0015 $\times$ 10^{+2}
F8	**8.5124** $\times$ **10**$^{+2}$	1.2534 $\times$ 10^{+3}	8.9569 $\times$ 10^{+2}	1.4234 $\times$ 10^{+3}	1.2436 $\times$ 10^{+3}	8.8733 $\times$ 10^{+2}
F9	**9.5224** $\times$ **10**$^{+2}$	1.3821 $\times$ 10^{+3}	1.0803 $\times$ 10^{+3}	1.6733 $\times$ 10^{+3}	1.3558 $\times$ 10^{+3}	1.0980 $\times$ 10^{+3}
F10	**3.2518** $\times$ **10**$^{+3}$	1.1643 $\times$ 10^{+4}	3.9794 $\times$ 10^{+3}	1.4554 $\times$ 10^{+4}	1.2651 $\times$ 10^{+4}	4.9779 $\times$ 10^{+3}
F11	**5.6490** $\times$ **10**$^{+3}$	1.3677 $\times$ 10^{+4}	1.4605 $\times$ 10^{+4}	1.5495 $\times$ 10^{+4}	1.4906 $\times$ 10^{+4}	1.3739 $\times$ 10^{+4}
F12	**1.2034** $\times$ **10**$^{+3}$	1.2036 $\times$ 10^{+3}	1.2041 $\times$ 10^{+3}	1.2039 $\times$ 10^{+3}	1.2042 $\times$ 10^{+3}	1.2043 $\times$ 10^{+3}

Table 5. *Cont.*

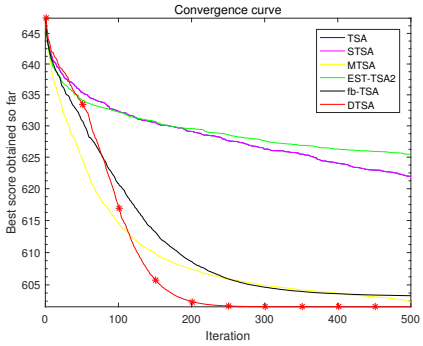
Function	DTSA	EST-TSA	MTSA	STSA	TSA	fb-TSA
F13	**1.3004** $\times$ **10**$^{+3}$	1.3008 $\times$ 10^{+3}	1.3006 $\times$ 10^{+3}	1.3071 $\times$ 10^{+3}	1.3012 $\times$ 10^{+3}	1.3006 $\times$ 10^{+3}
F14	**1.4004** $\times$ **10**$^{+3}$	1.4005 $\times$ 10^{+3}	1.4004 $\times$ 10^{+3}	1.7103 $\times$ 10^{+3}	1.4265 $\times$ 10^{+3}	1.4004 $\times$ 10^{+3}
F15	**1.5083** $\times$ **10**$^{+3}$	5.0779 $\times$ 10^{+3}	1.5372 $\times$ 10^{+3}	6.6525 $\times$ 10^{+6}	4.9848 $\times$ 10^{+3}	1.5313 $\times$ 10^{+3}
F16	**1.6183** $\times$ **10**$^{+3}$	1.6226 $\times$ 10^{+3}	1.6224 $\times$ 10^{+3}	1.6230 $\times$ 10^{+3}	1.6228 $\times$ 10^{+3}	1.6224 $\times$ 10^{+3}
F17	**1.0135** $\times$ **10**$^{+6}$	1.3693 $\times$ 10^{+7}	1.0388 $\times$ 10^{+6}	1.5580 $\times$ 10^{+8}	2.0814 $\times$ 10^{+7}	1.0716 $\times$ 10^{+6}
F18	**2.1716** $\times$ **10**$^{+3}$	3.6138 $\times$ 10^{+3}	3.1925 $\times$ 10^{+3}	2.5696 $\times$ 10^{+9}	1.0451 $\times$ 10^{+5}	2.5001 $\times$ 10^{+3}
F19	1.9680 $\times$ 10^{+3}	1.9831 $\times$ 10^{+3}	**1.9324** $\times$ **10**$^{+3}$	2.3253 $\times$ 10^{+3}	1.9917 $\times$ 10^{+3}	1.9371 $\times$ 10^{+3}
F20	**6.8302** $\times$ **10**$^{+3}$	4.2957 $\times$ 10^{+4}	1.5457 $\times$ 10^{+4}	2.7659 $\times$ 10^{+5}	4.8036 $\times$ 10^{+4}	3.7643 $\times$ 10^{+4}
F21	**4.4947** $\times$ **10**$^{+5}$	8.5454 $\times$ 10^{+6}	2.2217 $\times$ 10^{+6}	4.0869 $\times$ 10^{+7}	7.2602 $\times$ 10^{+6}	1.0551 $\times$ 10^{+6}
F22	**3.1275** $\times$ **10**$^{+3}$	3.6816 $\times$ 10^{+3}	3.6274 $\times$ 10^{+3}	5.2984 $\times$ 10^{+3}	3.9806 $\times$ 10^{+3}	3.7124 $\times$ 10^{+3}
F23	2.6440 $\times$ 10^{+3}	**2.5795** $\times$ **10**$^{+3}$	2.6441 $\times$ 10^{+3}	3.4661 $\times$ 10^{+3}	2.6688 $\times$ 10^{+3}	2.6440 $\times$ 10^{+3}
F24	2.6736 $\times$ 10^{+3}	**2.6000** $\times$ **10**$^{+3}$	2.6628 $\times$ 10^{+3}	2.8856 $\times$ 10^{+3}	2.7318 $\times$ 10^{+3}	2.6711 $\times$ 10^{+3}
F25	2.7209 $\times$ 10^{+3}	**2.7000** $\times$ **10**$^{+3}$	2.7168 $\times$ 10^{+3}	2.9058 $\times$ 10^{+3}	2.7741 $\times$ 10^{+3}	2.7325 $\times$ 10^{+3}
F26	2.7505 $\times$ 10^{+3}	2.8000 $\times$ 10^{+3}	2.7507 $\times$ 10^{+3}	2.7070 $\times$ 10^{+3}	**2.7017** $\times$ **10**$^{+3}$	2.8030 $\times$ 10^{+3}
F27	**3.1411** $\times$ **10**$^{+3}$	4.2662 $\times$ 10^{+3}	3.3051 $\times$ 10^{+3}	4.8809 $\times$ 10^{+3}	4.4235 $\times$ 10^{+3}	3.5434 $\times$ 10^{+3}
F28	4.2500 $\times$ 10^{+3}	6.3406 $\times$ 10^{+3}	**4.1564** $\times$ **10**$^{+3}$	8.8173 $\times$ 10^{+3}	6.0780 $\times$ 10^{+3}	4.8144 $\times$ 10^{+3}
F29	**4.3020** $\times$ **10**$^{+3}$	1.4916 $\times$ 10^{+6}	7.1278 $\times$ 10^{+3}	2.3249 $\times$ 10^{+8}	7.0369 $\times$ 10^{+6}	4.3833 $\times$ 10^{+3}
30	**1.6597** $\times$ **10**$^{+4}$	1.4608 $\times$ 10^{+5}	1.9499 $\times$ 10^{+4}	3.2828 $\times$ 10^{+6}	2.6339 $\times$ 10^{+5}	2.0468 $\times$ 10^{+4}
Rank first	24	3	2	0	1	1

Table 6. The mean values for the DTSA, EAT-TSA, fb-TSA, TSA, STSA, and MTSA in 100 dimensions.

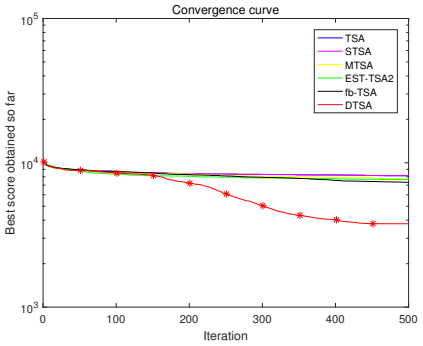
Function	DTSA	EST-TSA	MTSA	STSA	TSA	fb-TSA
F1	**5.7243** $\times$ **10**$^{+7}$	1.4918 $\times$ 10^{+9}	1.2921 $\times$ 10^{+8}	1.1868 $\times$ 10^{+10}	3.1047 $\times$ 10^{+9}	2.2997 $\times$ 10^{+8}
F2	**7.6447** $\times$ **10**$^{+4}$	6.2758 $\times$ 10^{+10}	2.6417 $\times$ 10^{+8}	4.6906 $\times$ 10^{+11}	8.1438 $\times$ 10^{+10}	1.9459 $\times$ 10^{+9}
F3	**2.8429** $\times$ **10**$^{+4}$	3.0542 $\times$ 10^{+5}	2.1455 $\times$ 10^{+5}	7.8482 $\times$ 10^{+5}	3.4631 $\times$ 10^{+5}	3.0294 $\times$ 10^{+5}
F4	**7.7869** $\times$ **10**$^{+2}$	1.0509 $\times$ 10^{+4}	8.7401 $\times$ 10^{+2}	1.5260 $\times$ 10^{+5}	1.3924 $\times$ 10^{+4}	1.0689 $\times$ 10^{+3}
F5	**5.2115** $\times$ **10**$^{+2}$	5.2137 $\times$ 10^{+2}	5.2138 $\times$ 10^{+2}	5.2134 $\times$ 10^{+2}	5.2138 $\times$ 10^{+2}	5.2137 $\times$ 10^{+2}
F6	**6.4302** $\times$ **10**$^{+2}$	7.3663 $\times$ 10^{+2}	6.6467 $\times$ 10^{+2}	7.6389 $\times$ 10^{+2}	7.4510 $\times$ 10^{+2}	6.7667 $\times$ 10^{+2}
F7	**7.0018** $\times$ **10**$^{+2}$	1.2792 $\times$ 10^{+3}	7.0693 $\times$ 10^{+2}	5.0465 $\times$ 10^{+3}	1.5563 $\times$ 10^{+3}	7.1093 $\times$ 10^{+2}
F8	**9.7213** $\times$ **10**$^{+2}$	1.9392 $\times$ 10^{+3}	1.1618 $\times$ 10^{+3}	2.6293 $\times$ 10^{+3}	1.9059 $\times$ 10^{+3}	1.1462 $\times$ 10^{+3}
F9	**1.0736** $\times$ **10**$^{+3}$	2.1834 $\times$ 10^{+3}	1.7852 $\times$ 10^{+3}	2.9956 $\times$ 10^{+3}	2.0958 $\times$ 10^{+3}	1.5184 $\times$ 10^{+3}
F10	**7.0297** $\times$ **10**$^{+3}$	2.9105 $\times$ 10^{+4}	1.8204 $\times$ 10^{+4}	3.2641 $\times$ 10^{+4}	3.0755 $\times$ 10^{+4}	1.4705 $\times$ 10^{+4}
F11	**1.4719** $\times$ **10**$^{+4}$	3.0524 $\times$ 10^{+4}	3.1888 $\times$ 10^{+4}	3.2506 $\times$ 10^{+4}	3.2241 $\times$ 10^{+4}	3.1913 $\times$ 10^{+4}
F12	**1.2024** $\times$ **10**$^{+3}$	1.2043 $\times$ 10^{+3}	1.2045 $\times$ 10^{+3}	1.2045 $\times$ 10^{+3}	1.2049 $\times$ 10^{+3}	1.2046 $\times$ 10^{+3}
F13	**1.3007** $\times$ **10**$^{+3}$	1.3040 $\times$ 10^{+3}	1.3008 $\times$ 10^{+3}	1.3120 $\times$ 10^{+3}	1.3049 $\times$ 10^{+3}	1.3009 $\times$ 10^{+3}
F14	**1.4004** $\times$ **10**$^{+3}$	1.5730 $\times$ 10^{+3}	1.4007 $\times$ 10^{+3}	2.7335 $\times$ 10^{+3}	1.6373 $\times$ 10^{+3}	1.4004 $\times$ 10^{+3}
F15	**1.5510** $\times$ **10**$^{+3}$	4.8840 $\times$ 10^{+5}	1.6074 $\times$ 10^{+3}	2.1651 $\times$ 10^{+8}	1.2171 $\times$ 10^{+6}	2.0121 $\times$ 10^{+3}
F16	**1.6448** $\times$ **10**$^{+3}$	1.6470 $\times$ 10^{+3}	1.6467 $\times$ 10^{+3}	1.6477 $\times$ 10^{+3}	1.6470 $\times$ 10^{+3}	1.6470 $\times$ 10^{+3}
F17	**3.6189** $\times$ **10**$^{+6}$	1.6916 $\times$ 10^{+8}	1.6867 $\times$ 10^{+7}	1.1497 $\times$ 10^{+9}	2.7832 $\times$ 10^{+8}	3.9192 $\times$ 10^{+7}
F18	**2.3118** $\times$ **10**$^{+3}$	4.0168 $\times$ 10^{+4}	5.9916 $\times$ 10^{+4}	2.5461 $\times$ 10^{+10}	2.0551 $\times$ 10^{+8}	3.0225 $\times$ 10^{+3}
F19	2.0177 $\times$ 10^{+3}	2.2319 $\times$ 10^{+3}	2.0154 $\times$ 10^{+3}	5.9575 $\times$ 10^{+3}	2.3520 $\times$ 10^{+3}	**2.0079** $\times$ **10**$^{+3}$
F20	**3.5926** $\times$ **10**$^{+4}$	2.5544 $\times$ 10^{+5}	8.3978 $\times$ 10^{+4}	3.2150 $\times$ 10^{+6}	3.0385 $\times$ 10^{+5}	2.0031 $\times$ 10^{+5}
F21	**2.6598** $\times$ **10**$^{+6}$	6.2281 $\times$ 10^{+7}	7.2953 $\times$ 10^{+6}	4.8294 $\times$ 10^{+8}	1.0373 $\times$ 10^{+8}	7.2459 $\times$ 10^{+6}
F22	**4.0180** $\times$ **10**$^{+3}$	6.9075 $\times$ 10^{+3}	5.2165 $\times$ 10^{+3}	9.8067 $\times$ 10^{+3}	6.7491 $\times$ 10^{+3}	6.9634 $\times$ 10^{+3}
F23	2.6485 $\times$ 10^{+3}	**2.5000** $\times$ **10**$^{+3}$	2.6555 $\times$ 10^{+3}	5.9704 $\times$ 10^{+3}	2.9402 $\times$ 10^{+3}	2.6556 $\times$ 10^{+3}
F24	2.7984 $\times$ 10^{+3}	**2.6000** $\times$ **10**$^{+3}$	2.7736 $\times$ 10^{+3}	3.9225 $\times$ 10^{+3}	3.0772 $\times$ 10^{+3}	2.8196 $\times$ 10^{+3}
F25	2.7765 $\times$ 10^{+3}	**2.7000** $\times$ **10**$^{+3}$	2.7716 $\times$ 10^{+3}	3.8529 $\times$ 10^{+3}	3.0918 $\times$ 10^{+3}	2.7956 $\times$ 10^{+3}
F26	2.8014 $\times$ 10^{+3}	2.8000 $\times$ 10^{+3}	2.8031 $\times$ 10^{+3}	**2.7185** $\times$ **10**$^{+3}$	3.0466 $\times$ 10^{+3}	2.8094 $\times$ 10^{+3}
F27	**3.9995** $\times$ **10**$^{+3}$	6.3806 $\times$ 10^{+3}	4.5261 $\times$ 10^{+3}	7.4034 $\times$ 10^{+3}	6.5736 $\times$ 10^{+3}	5.0424 $\times$ 10^{+3}
F28	**6.3978** $\times$ **10**$^{+3}$	2.2158 $\times$ 10^{+4}	7.0157 $\times$ 10^{+3}	2.2799 $\times$ 10^{+4}	1.9432 $\times$ 10^{+4}	1.2124 $\times$ 10^{+4}
F29	**6.7693** $\times$ **10**$^{+3}$	4.9012 $\times$ 10^{+7}	1.2955 $\times$ 10^{+5}	1.5016 $\times$ 10^{+9}	1.6496 $\times$ 10^{+8}	1.8635 $\times$ 10^{+4}
F30	**3.7315** $\times$ **10**$^{+4}$	3.2730 $\times$ 10^{+6}	1.5121 $\times$ 10^{+5}	6.5268 $\times$ 10^{+7}	7.2847 $\times$ 10^{+6}	1.1320 $\times$ 10^{+5}
Rank first	25	3	0	1	0	1



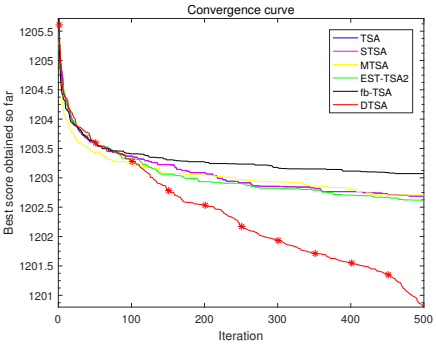
(**a**) f_3, unimodal function



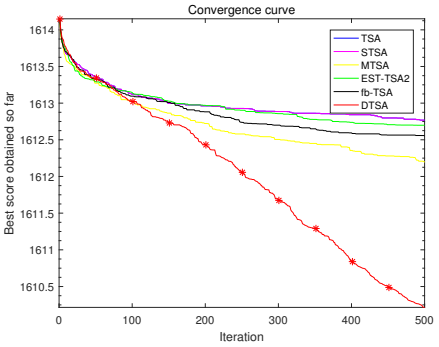
(**b**) f_6, multimodal function



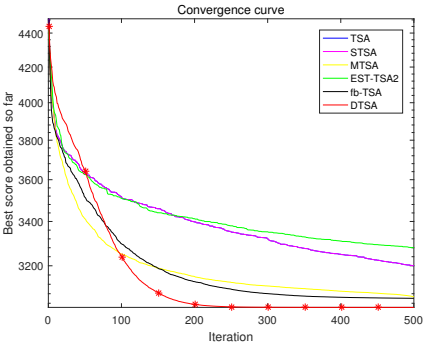
(**c**) f_{10}, multimodal function



(**d**) f_{17}, multimodal function

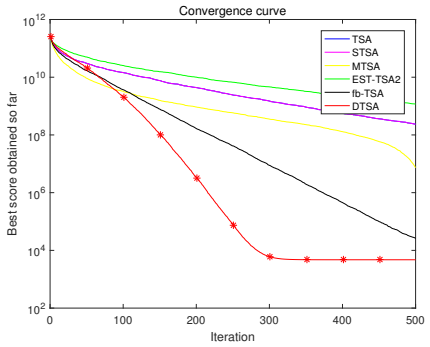


(**e**) f_{24}, hybrid function

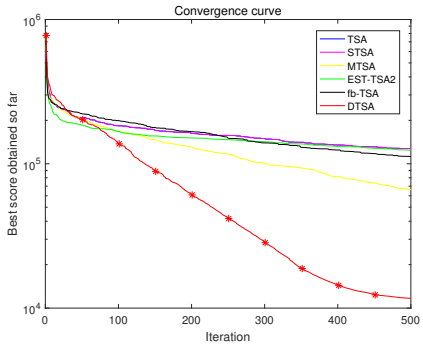


(**f**) f_{25}, composite function

Figure 15. Convergence curves of the DTSA and TSA and its recent variants in 30 dimensions.

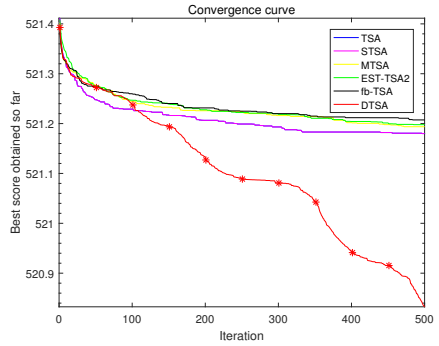


(**a**) f_3, unimodal function

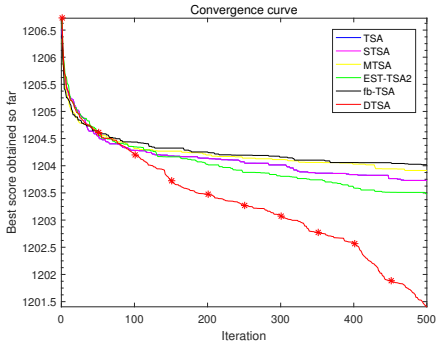


(**b**) f_6, multimodal function

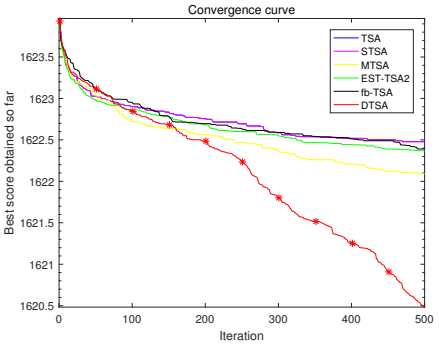
Figure 16. *Cont.*



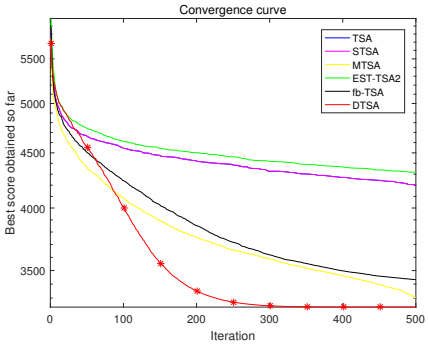
(**c**) f_{10}, multimodal function



(**d**) f_{17}, multimodal function

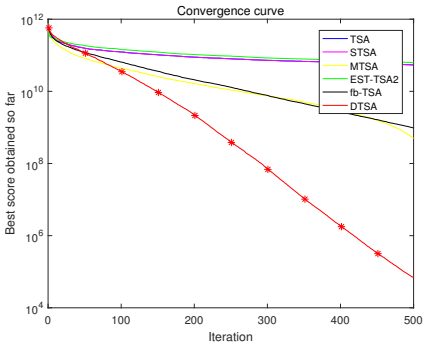


(**e**) f_{24}, hybrid function

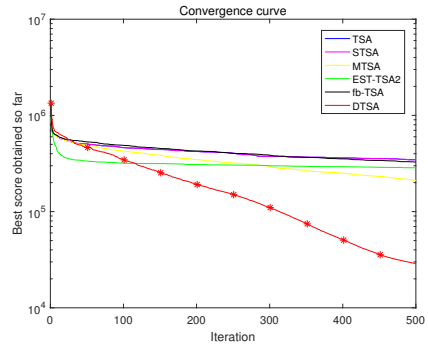


(**f**) f_{25}, composite function

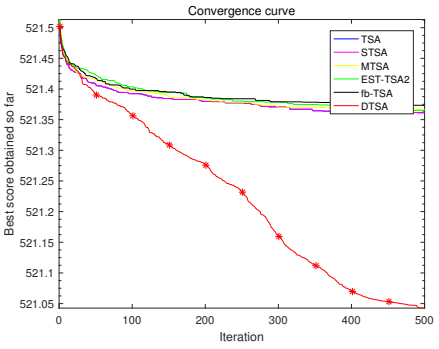
Figure 16. Convergence curves of the DTSA and TSA and its recent variants in 50 dimensions.



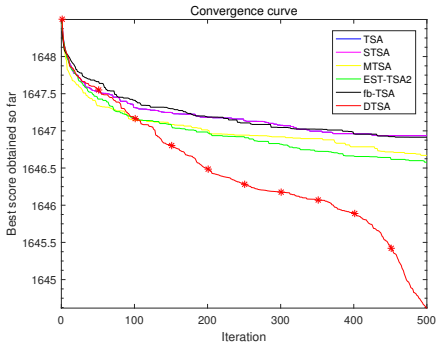
(**a**) f_3, unimodal function



(**b**) f_6, multimodal function

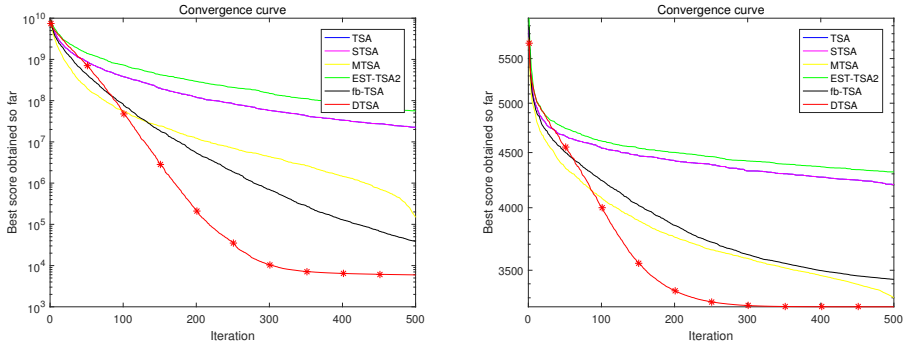


(**c**) f_{10}, multimodal function



(**d**) f_{17}, multimodal function

Figure 17. *Cont.*



(**e**) f_{24}, hybrid function (**f**) f_{25}, composite function

Figure 17. Convergence curve of the DTSA and TSA and its recent variants in 100 dimensions.

4.3.2. Comparative Experiment 2: DTSA versus Classical and Recent Swarm Intelligence Algorithms

To substantiate the efficacy of the proposed algorithm, additional experiments were meticulously conducted. Classical swarm intelligence algorithms, including GA [58] and PSO [63], and some recent and famous swarm intelligence algorithms, such as RSA [64] and BOA [65], were deliberately chosen for comparative analyses.

The experimental setup rigorously adhered to the specifications outlined in Section 4.3.1, comprising 30 rounds of experiments with 500 iterations per round. Problem dimensions varied across 30, 50, and 100, maintaining consistency with earlier experiments. Furthermore, parameters for each algorithm were meticulously configured according to the settings outlined in their respective original papers, as delineated in Table 1.

Tables 7–9 document the average best values attained by DTSA and its counterparts across 30 rounds of experimentation, providing insights into the average rankings of each algorithm. These tables validate DTSA's robust performance, particularly in addressing multimodal problems. Additionally, graphical depictions of convergence curves for these algorithms are illustrated in Figures 18–20.

Table 7. The mean values for the DTSA, EAT-TSA, fb_TSA, TSA, STSA, MTSA, PSO, GWO, BOA and RSA in 30 dimensions.

Function	DTSA	TSA	STSA	MTSA	EST-TSA	fb-TSA	PSO	GWO	BOA	RSA	GA
Unimodal functions	**1.65** × **10**$^{+6}$	9.39 × 10^{+7}	5.92 × 10^{+8}	5.92 × 10^{+6}	1.04 × 10^{+8}	8.07 × 10^{+6}	3.84 × 10^{+7}	1.32 × 10^{+8}	1.64 × 10^{+9}	1.05 × 10^{+9}	6.05 × 10^{+8}
	5.80 × 10^{+3}	3.93 × 10^{+5}	2.63 × 10^{+10}	2.24 × 10^{+5}	3.10 × 10^{+6}	**5.59** × **10**$^{+3}$	4.42 × 10^{+6}	5.36 × 10^{+9}	7.75 × 10^{+10}	7.18 × 10^{+10}	3.62 × 10^{+10}
	4.48 × **10**$^{+3}$	3.74 × 10^{+4}	8.66 × 10^{+4}	4.01 × 10^{+3}	3.94 × 10^{+4}	1.19 × 10^{+4}	3.29 × 10^{+4}	5.64 × 10^{+4}	7.61 × 10^{+4}	7.95 × 10^{+4}	7.14 × 10^{+4}
Simple multimodal functions	5.00 × 10^{+2}	5.53 × 10^{+2}	2.89 × 10^{+3}	5.01 × 10^{+2}	5.91 × 10^{+2}	**4.96** × **10**$^{+2}$	6.44 × 10^{+2}	8.03 × 10^{+2}	1.64 × 10^{+4}	8.78 × 10^{+3}	5.48 × 10^{+3}
	5.21 × **10**$^{+2}$	5.21 × 10^{+2}	5.21 × 10^{+2}	5.21 × 10^{+2}	5.21 × 10^{+2}	5.21 × 10^{+2}	5.21 × 10^{+2}	5.21 × 10^{+2}	5.21 × 10^{+2}	5.21 × 10^{+2}	5.21 × 10^{+2}
	6.02 × **10**$^{+2}$	6.23 × 10^{+2}	6.39 × 10^{+2}	6.03 × 10^{+2}	6.26 × 10^{+2}	6.03 × 10^{+2}	6.18 × 10^{+2}	6.19 × 10^{+2}	6.38 × 10^{+2}	6.40 × 10^{+2}	6.37 × 10^{+2}
	7.00 × **10**$^{+2}$	7.00 × 10^{+2}	9.50 × 10^{+2}	7.00 × 10^{+2}	7.00 × 10^{+2}	7.00 × 10^{+2}	7.01 × 10^{+2}	7.54 × 10^{+2}	1.47 × 10^{+3}	1.30 × 10^{+3}	1.07 × 10^{+3}
	8.23 × **10**$^{+2}$	9.68 × 10^{+2}	1.08 × 10^{+3}	8.31 × 10^{+2}	9.91 × 10^{+2}	8.37 × 10^{+2}	8.57 × 10^{+2}	9.10 × 10^{+2}	1.12 × 10^{+3}	1.16 × 10^{+3}	1.05 × 10^{+3}
	9.32 × **10**$^{+2}$	1.11 × 10^{+3}	1.20 × 10^{+3}	9.75 × 10^{+2}	1.14 × 10^{+3}	9.77 × 10^{+2}	1.03 × 10^{+3}	1.04 × 10^{+3}	1.25 × 10^{+3}	1.24 × 10^{+3}	1.17 × 10^{+3}
	1.60 × **10**$^{+3}$	6.28 × 10^{+3}	7.74 × 10^{+3}	1.96 × 10^{+3}	6.02 × 10^{+3}	2.20 × 10^{+3}	3.01 × 10^{+3}	4.51 × 10^{+3}	8.58 × 10^{+3}	7.92 × 10^{+3}	7.17 × 10^{+3}
	3.79 × **10**$^{+3}$	8.18 × 10^{+3}	8.45 × 10^{+3}	7.49 × 10^{+3}	7.53 × 10^{+3}	7.80 × 10^{+3}	7.30 × 10^{+3}	5.81 × 10^{+3}	8.98 × 10^{+3}	8.81 × 10^{+3}	7.99 × 10^{+3}
	1.20 × **10**$^{+3}$	1.20 × 10^{+3}	1.20 × 10^{+3}	1.20 × 10^{+3}	1.20 × 10^{+3}	1.20 × 10^{+3}	1.20 × 10^{+3}	1.20 × 10^{+3}	1.20 × 10^{+3}	1.20 × 10^{+3}	1.20 × 10^{+3}
	1.30 × **10**$^{+3}$	1.30 × 10^{+3}	1.30 × 10^{+3}	1.30 × 10^{+3}	1.30 × 10^{+3}	1.30 × 10^{+3}	1.30 × 10^{+3}	1.30 × 10^{+3}	1.31 × 10^{+3}	1.31 × 10^{+3}	1.31 × 10^{+3}
	1.40 × **10**$^{+3}$	1.40 × 10^{+3}	1.48 × 10^{+3}	1.40 × 10^{+3}	1.40 × 10^{+3}	1.40 × 10^{+3}	1.40 × 10^{+3}	1.42 × 10^{+3}	1.71 × 10^{+3}	1.57 × 10^{+3}	1.55 × 10^{+3}
	1.50 × **10**$^{+3}$	1.52 × 10^{+3}	1.37 × 10^{+4}	1.52 × 10^{+3}	1.52 × 10^{+3}	1.51 × 10^{+3}	1.52 × 10^{+3}	2.36 × 10^{+3}	3.98 × 10^{+5}	2.0 × 10^{+5}	1.71 × 10^{+4}
	1.61 × **10**$^{+3}$	1.61 × 10^{+3}	1.61 × 10^{+3}	1.61 × 10^{+3}	1.61 × 10^{+3}	1.61 × 10^{+3}	1.61 × 10^{+3}	1.61 × 10^{+3}	1.61 × 10^{+3}	1.61 × 10^{+3}	1.61 × 10^{+3}
Hybrid functions	**3.83** × **10**$^{+5}$	2.32 × 10^{+6}	1.50 × 10^{+7}	5.30 × 10^{+5}	2.04 × 10^{+6}	5.78 × 10^{+5}	2.60 × 10^{+6}	5.95 × 10^{+6}	1.91 × 10^{+8}	1.08 × 10^{+8}	3.38 × 10^{+7}
	2.62 × 10^{+3}	2.32 × 10^{+3}	1.19 × 10^{+8}	2.81 × 10^{+3}	**2.21** × **10**$^{+3}$	2.22 × 10^{+3}	7.05 × 10^{+5}	1.97 × 10^{+7}	6.28 × 10^{+9}	4.51 × 10^{+9}	7.71 × 10^{+8}
	1.90 × **10**$^{+3}$	1.91 × 10^{+3}	2.00 × 10^{+3}	1.91 × 10^{+3}	1.91 × 10^{+3}	1.91 × 10^{+3}	1.92 × 10^{+3}	1.94 × 10^{+3}	2.44 × 10^{+3}	2.25 × 10^{+3}	2.17 × 10^{+3}
	7.62 × **10**$^{+3}$	1.62 × 10^{+4}	5.49 × 10^{+4}	7.66 × 10^{+3}	2.21 × 10^{+4}	1.44 × 10^{+4}	3.39 × 10^{+4}	5.10 × 10^{+4}	3.70 × 10^{+5}	1.60 × 10^{+5}	6.39 × 10^{+4}
	1.37 × **10**$^{+5}$	4.03 × 10^{+5}	3.14 × 10^{+6}	1.53 × 10^{+5}	4.84 × 10^{+5}	1.52 × 10^{+5}	4.80 × 10^{+5}	2.05 × 10^{+6}	5.24 × 10^{+7}	4.28 × 10^{+7}	7.92 × 10^{+6}
	2.41 × 10^{+3}	2.69 × 10^{+3}	3.24 × 10^{+3}	**2.37** × **10**$^{+3}$	2.68 × 10^{+3}	2.42 × 10^{+3}	2.60 × 10^{+3}	2.76 × 10^{+3}	3.48 × 10^{+4}	2.06 × 10^{+4}	3.53 × 10^{+3}
	2.62 × 10^{+3}	2.62 × 10^{+3}	2.72 × 10^{+3}	2.62 × 10^{+3}	2.60 × 10^{+3}	2.62 × 10^{+3}	2.62 × 10^{+3}	2.65 × 10^{+3}	**2.50** × **10**$^{+3}$	2.50 × 10^{+3}	2.79 × 10^{+3}

Table 7. *Cont.*

Function	DTSA	TSA	STSA	MTSA	EST-TSA	fb-TSA	PSO	GWO	BOA	RSA	GA
Composition functions	$2.63 \times 10^{+3}$	$2.63 \times 10^{+3}$	$2.60 \times 10^{+3}$	$2.62 \times 10^{+3}$	$2.60 \times 10^{+3}$	$2.63 \times 10^{+3}$	$2.64 \times 10^{+3}$	$2.60 \times 10^{+3}$	$2.60 \times 10^{+3}$	$\mathbf{2.60 \times 10^{+3}}$	$2.62 \times 10^{+3}$
	$2.71 \times 10^{+3}$	$2.72 \times 10^{+3}$	$2.75 \times 10^{+3}$	$2.71 \times 10^{+3}$	$2.70 \times 10^{+3}$	$2.71 \times 10^{+3}$	$2.71 \times 10^{+3}$	$2.70 \times 10^{+3}$	$\mathbf{2.70 \times 10^{+3}}$	$2.70 \times 10^{+3}$	$2.71 \times 10^{+3}$
	$\mathbf{2.70 \times 10^{+3}}$	$2.70 \times 10^{+3}$	$2.70 \times 10^{+3}$	$2.71 \times 10^{+3}$	$2.77 \times 10^{+3}$	$2.70 \times 10^{+3}$	$2.74 \times 10^{+3}$	$2.72 \times 10^{+3}$	$2.77 \times 10^{+3}$	$2.80 \times 10^{+3}$	$2.72 \times 10^{+3}$
	$\mathbf{3.04 \times 10^{+3}}$	$3.22 \times 10^{+3}$	$3.72 \times 10^{+3}$	$3.06 \times 10^{+3}$	$3.30 \times 10^{+3}$	$3.06 \times 10^{+3}$	$3.40 \times 10^{+3}$	$3.55 \times 10^{+3}$	$3.56 \times 10^{+3}$	$4.20 \times 10^{+3}$	$3.74 \times 10^{+3}$
	$3.71 \times 10^{+3}$	$4.04 \times 10^{+3}$	$5.29 \times 10^{+3}$	$3.67 \times 10^{+3}$	$4.09 \times 10^{+3}$	$3.71 \times 10^{+3}$	$4.65 \times 10^{+3}$	$\mathbf{3.35 \times 10^{+3}}$	$5.84 \times 10^{+3}$	$5.73 \times 10^{+3}$	$8.72 \times 10^{+3}$
	$3.97 \times 10^{+3}$	$4.06 \times 10^{+4}$	$2.32 \times 10^{+7}$	$3.97 \times 10^{+3}$	$4.84 \times 10^{+4}$	$3.94 \times 10^{+3}$	$4.42 \times 10^{+6}$	$\mathbf{3.11 \times 10^{+3}}$	$9.16 \times 10^{+6}$	$1.63 \times 10^{+7}$	$2.98 \times 10^{+8}$
	$4.51 \times 10^{+3}$	$1.27 \times 10^{+4}$	$4.00 \times 10^{+5}$	$4.99 \times 10^{+3}$	$1.84 \times 10^{+4}$	$5.07 \times 10^{+3}$	$1.99 \times 10^{+4}$	$\mathbf{3.56 \times 10^{+3}}$	$3.51 \times 10^{+6}$	$2.49 \times 10^{+6}$	$1.37 \times 10^{+6}$
Ranking first	20	0	0	1	1	2	0	3	2	1	0

Table 8. The mean values for the DTSA, EAT-TSA, fb_TSA, TSA, STSA, MTSA, PSO, GWO, BOA and RSA in 50 dimensions.

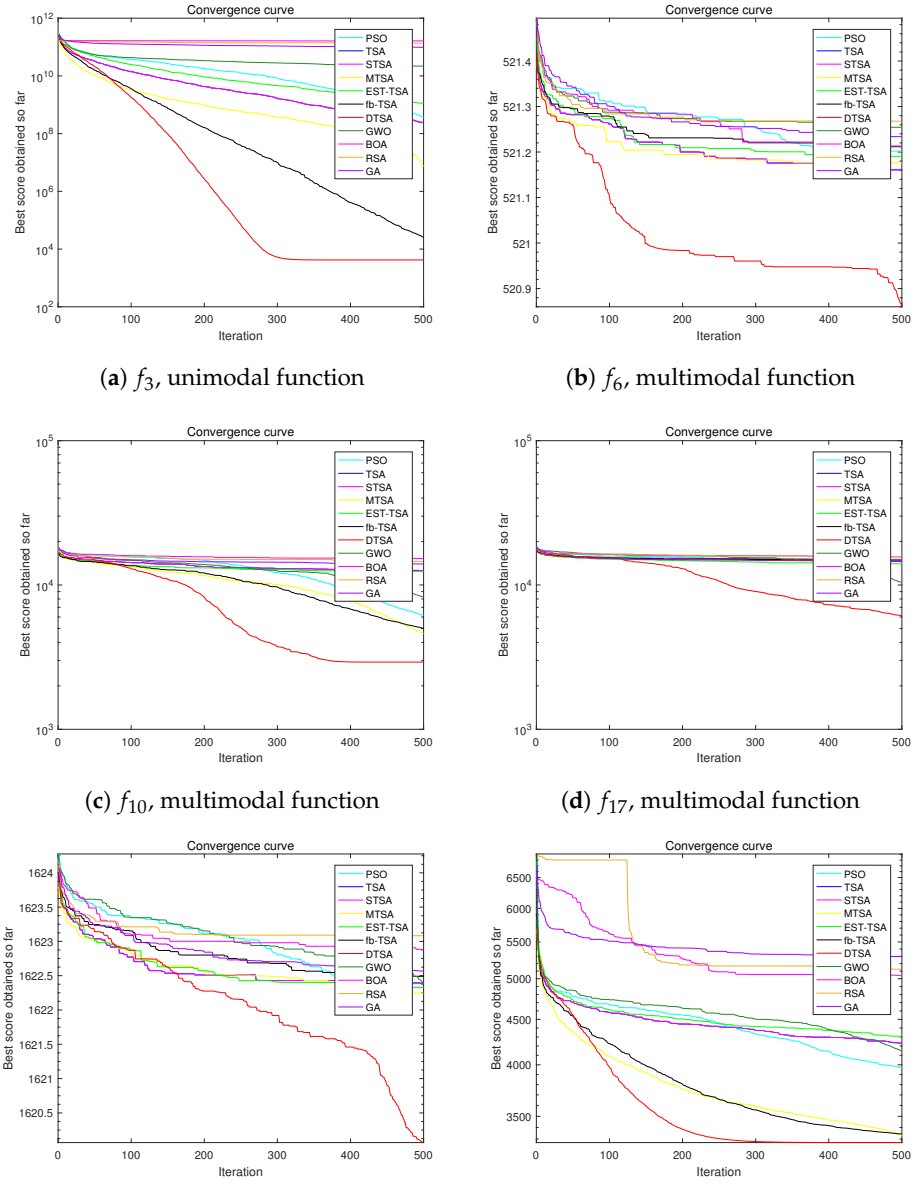
Function	DTSA	TSA	STSA	MTSA	EST-TSA	fb-TSA	PSO	GWO	BOA	RSA	GA
Unimodal functions	$\mathbf{3.06 \times 10^{+6}}$	$3.56 \times 10^{+8}$	$2.35 \times 10^{+9}$	$8.95 \times 10^{+6}$	$2.99 \times 10^{+8}$	$4.43 \times 10^{+7}$	$9.03 \times 10^{+7}$	$4.72 \times 10^{+8}$	$5.45 \times 10^{+9}$	$2.93 \times 10^{+9}$	$2.43 \times 10^{+9}$
	$\mathbf{2.34 \times 10^{+3}}$	$2.45 \times 10^{+8}$	$1.15 \times 10^{+11}$	$8.51 \times 10^{+6}$	$1.07 \times 10^{+9}$	$2.10 \times 10^{+4}$	$2.44 \times 10^{+8}$	$1.66 \times 10^{+10}$	$1.79 \times 10^{+11}$	$1.40 \times 10^{+11}$	$9.68 \times 10^{+10}$
	$\mathbf{1.36 \times 10^{+4}}$	$1.19 \times 10^{+5}$	$2.98 \times 10^{+5}$	$7.72 \times 10^{+4}$	$1.10 \times 10^{+5}$	$1.33 \times 10^{+5}$	$1.45 \times 10^{+5}$	$1.66 \times 10^{+5}$	$1.99 \times 10^{+5}$	$1.48 \times 10^{+5}$	$1.43 \times 10^{+5}$
Simple multimodal functions	$\mathbf{5.15 \times 10^{+2}}$	$8.83 \times 10^{+2}$	$2.84 \times 10^{+4}$	$5.40 \times 10^{+2}$	$1.26 \times 10^{+3}$	$5.36 \times 10^{+2}$	$7.50 \times 10^{+2}$	$2.91 \times 10^{+3}$	$5.13 \times 10^{+4}$	$2.51 \times 10^{+4}$	$2.33 \times 10^{+4}$
	$\mathbf{5.21 \times 10^{+2}}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$
	$\mathbf{6.07 \times 10^{+2}}$	$6.54 \times 10^{+2}$	$6.74 \times 10^{+2}$	$6.08 \times 10^{+2}$	$6.54 \times 10^{+2}$	$6.16 \times 10^{+2}$	$6.42 \times 10^{+2}$	$6.39 \times 10^{+2}$	$6.68 \times 10^{+2}$	$6.75 \times 10^{+2}$	$6.68 \times 10^{+2}$
	$\mathbf{7.00 \times 10^{+2}}$	$7.02 \times 10^{+2}$	$1.81 \times 10^{+3}$	$7.01 \times 10^{+2}$	$7.12 \times 10^{+2}$	$7.00 \times 10^{+2}$	$7.04 \times 10^{+2}$	$9.23 \times 10^{+2}$	$2.30 \times 10^{+3}$	$1.90 \times 10^{+3}$	$1.64 \times 10^{+3}$
	$\mathbf{8.50 \times 10^{+2}}$	$1.21 \times 10^{+3}$	$1.44 \times 10^{+3}$	$8.83 \times 10^{+2}$	$1.23 \times 10^{+3}$	$8.98 \times 10^{+2}$	$9.26 \times 10^{+2}$	$1.05 \times 10^{+3}$	$1.43 \times 10^{+3}$	$1.48 \times 10^{+3}$	$1.28 \times 10^{+3}$
	$\mathbf{9.74 \times 10^{+2}}$	$1.37 \times 10^{+3}$	$1.65 \times 10^{+3}$	$1.09 \times 10^{+3}$	$1.40 \times 10^{+3}$	$1.15 \times 10^{+3}$	$1.16 \times 10^{+3}$	$1.21 \times 10^{+3}$	$1.60 \times 10^{+3}$	$1.61 \times 10^{+3}$	$1.56 \times 10^{+3}$
	$\mathbf{2.24 \times 10^{+3}}$	$1.30 \times 10^{+4}$	$1.51 \times 10^{+4}$	$4.48 \times 10^{+3}$	$1.23 \times 10^{+4}$	$5.23 \times 10^{+3}$	$6.37 \times 10^{+3}$	$8.41 \times 10^{+3}$	$1.56 \times 10^{+4}$	$1.47 \times 10^{+4}$	$1.30 \times 10^{+4}$
	$\mathbf{5.32 \times 10^{+3}}$	$1.45 \times 10^{+4}$	$1.53 \times 10^{+4}$	$1.45 \times 10^{+4}$	$1.41 \times 10^{+4}$	$1.48 \times 10^{+4}$	$1.23 \times 10^{+4}$	$8.52 \times 10^{+3}$	$1.59 \times 10^{+4}$	$1.53 \times 10^{+4}$	$1.43 \times 10^{+4}$
	$\mathbf{1.20 \times 10^{+3}}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$
	$\mathbf{1.30 \times 10^{+3}}$	$1.30 \times 10^{+3}$	$1.31 \times 10^{+3}$	$1.30 \times 10^{+3}$	$1.30 \times 10^{+3}$	$1.30 \times 10^{+3}$	$1.30 \times 10^{+3}$	$1.30 \times 10^{+3}$	$1.31 \times 10^{+3}$	$1.31 \times 10^{+3}$	$1.31 \times 10^{+3}$
	$\mathbf{1.40 \times 10^{+3}}$	$1.40 \times 10^{+3}$	$1.70 \times 10^{+3}$	$1.40 \times 10^{+3}$	$1.40 \times 10^{+3}$	$1.40 \times 10^{+3}$	$1.40 \times 10^{+3}$	$1.46 \times 10^{+3}$	$1.81 \times 10^{+3}$	$1.72 \times 10^{+3}$	$1.63 \times 10^{+3}$
	$\mathbf{1.51 \times 10^{+3}}$	$1.78 \times 10^{+3}$	$3.09 \times 10^{+6}$	$1.54 \times 10^{+3}$	$3.46 \times 10^{+3}$	$1.53 \times 10^{+3}$	$1.57 \times 10^{+3}$	$7.03 \times 10^{+4}$	$5.77 \times 10^{+6}$	$1.72 \times 10^{+6}$	$5.12 \times 10^{+5}$
	$\mathbf{1.62 \times 10^{+3}}$	$1.62 \times 10^{+3}$	$1.62 \times 10^{+3}$	$1.62 \times 10^{+3}$	$1.62 \times 10^{+3}$	$1.62 \times 10^{+3}$	$1.62 \times 10^{+3}$	$1.62 \times 10^{+3}$	$1.62 \times 10^{+3}$	$1.62 \times 10^{+3}$	$1.62 \times 10^{+3}$
Hybrid functions	$\mathbf{5.27 \times 10^{+5}}$	$1.56 \times 10^{+7}$	$1.14 \times 10^{+8}$	$2.27 \times 10^{+6}$	$2.45 \times 10^{+7}$	$1.76 \times 10^{+6}$	$1.31 \times 10^{+7}$	$1.85 \times 10^{+7}$	$9.37 \times 10^{+8}$	$3.85 \times 10^{+8}$	$1.75 \times 10^{+8}$
	$2.29 \times 10^{+3}$	$2.76 \times 10^{+3}$	$2.35 \times 10^{+9}$	$2.49 \times 10^{+3}$	$3.00 \times 10^{+3}$	$\mathbf{2.13 \times 10^{+3}}$	$2.30 \times 10^{+3}$	$4.37 \times 10^{+7}$	$2.46 \times 10^{+10}$	$1.05 \times 10^{+10}$	$5.29 \times 10^{+9}$
	$1.93 \times 10^{+3}$	$1.98 \times 10^{+3}$	$2.34 \times 10^{+3}$	$1.95 \times 10^{+3}$	$1.99 \times 10^{+3}$	$\mathbf{1.92 \times 10^{+3}}$	$1.97 \times 10^{+3}$	$2.03 \times 10^{+3}$	$6.52 \times 10^{+3}$	$4.17 \times 10^{+3}$	$2.70 \times 10^{+3}$
	$\mathbf{1.18 \times 10^{+4}}$	$3.66 \times 10^{+4}$	$4.29 \times 10^{+5}$	$2.61 \times 10^{+4}$	$3.11 \times 10^{+4}$	$5.02 \times 10^{+4}$	$7.18 \times 10^{+4}$	$2.05 \times 10^{+5}$	$2.71 \times 10^{+5}$	$1.60 \times 10^{+5}$	$6.96 \times 10^{+4}$
	$\mathbf{1.20 \times 10^{+6}}$	$6.05 \times 10^{+6}$	$3.83 \times 10^{+7}$	$1.51 \times 10^{+6}$	$4.30 \times 10^{+6}$	$\mathbf{5.56 \times 10^{+5}}$	$8.61 \times 10^{+6}$	$7.82 \times 10^{+6}$	$1.07 \times 10^{+8}$	$5.64 \times 10^{+7}$	$1.90 \times 10^{+7}$
	$\mathbf{2.82 \times 10^{+3}}$	$3.94 \times 10^{+3}$	$5.23 \times 10^{+3}$	$3.01 \times 10^{+3}$	$3.92 \times 10^{+3}$	$3.34 \times 10^{+3}$	$3.95 \times 10^{+3}$	$3.53 \times 10^{+3}$	$6.75 \times 10^{+5}$	$1.21 \times 10^{+6}$	$7.32 \times 10^{+3}$
	$2.64 \times 10^{+3}$	$2.65 \times 10^{+3}$	$3.38 \times 10^{+3}$	$2.64 \times 10^{+3}$	$2.58 \times 10^{+3}$	$2.64 \times 10^{+3}$	$2.66 \times 10^{+3}$	$2.83 \times 10^{+3}$	$\mathbf{2.50 \times 10^{+3}}$	$2.50 \times 10^{+3}$	$2.89 \times 10^{+3}$
Composition functions	$2.67 \times 10^{+3}$	$2.70 \times 10^{+3}$	$2.91 \times 10^{+3}$	$2.67 \times 10^{+3}$	$2.60 \times 10^{+3}$	$2.67 \times 10^{+3}$	$2.70 \times 10^{+3}$	$2.64 \times 10^{+3}$	$2.60 \times 10^{+3}$	$\mathbf{2.60 \times 10^{+3}}$	$2.66 \times 10^{+3}$
	$2.72 \times 10^{+3}$	$2.78 \times 10^{+3}$	$2.87 \times 10^{+3}$	$2.71 \times 10^{+3}$	$\mathbf{2.70 \times 10^{+3}}$	$2.72 \times 10^{+3}$	$2.73 \times 10^{+3}$	$2.71 \times 10^{+3}$	$\mathbf{2.70 \times 10^{+3}}$	$\mathbf{2.70 \times 10^{+3}}$	$2.71 \times 10^{+3}$
	$\mathbf{2.80 \times 10^{+3}}$	$2.77 \times 10^{+3}$	$\mathbf{2.71 \times 10^{+3}}$	$2.80 \times 10^{+3}$	$2.80 \times 10^{+3}$	$2.75 \times 10^{+3}$	$2.90 \times 10^{+3}$	$2.86 \times 10^{+3}$	$2.80 \times 10^{+3}$	$2.80 \times 10^{+3}$	$2.80 \times 10^{+3}$
	$\mathbf{3.20 \times 10^{+3}}$	$4.11 \times 10^{+3}$	$4.98 \times 10^{+3}$	$3.29 \times 10^{+3}$	$4.29 \times 10^{+3}$	$3.56 \times 10^{+3}$	$4.10 \times 10^{+3}$	$4.11 \times 10^{+3}$	$5.40 \times 10^{+3}$	$5.11 \times 10^{+3}$	$5.31 \times 10^{+3}$
	$4.16 \times 10^{+3}$	$5.70 \times 10^{+3}$	$9.86 \times 10^{+3}$	$4.24 \times 10^{+3}$	$8.31 \times 10^{+3}$	$4.33 \times 10^{+3}$	$6.48 \times 10^{+3}$	$\mathbf{3.51 \times 10^{+3}}$	$1.48 \times 10^{+4}$	$1.20 \times 10^{+4}$	$1.59 \times 10^{+4}$
	$3.70 \times 10^{+3}$	$8.68 \times 10^{+5}$	$2.99 \times 10^{+8}$	$6.75 \times 10^{+3}$	$1.57 \times 10^{+6}$	$5.16 \times 10^{+3}$	$1.12 \times 10^{+5}$	$\mathbf{3.12 \times 10^{+3}}$	$\mathbf{3.10 \times 10^{+3}}$	$\mathbf{3.10 \times 10^{+3}}$	$1.09 \times 10^{+9}$
	$1.33 \times 10^{+4}$	$1.40 \times 10^{+5}$	$3.19 \times 10^{+6}$	$1.84 \times 10^{+4}$	$2.08 \times 10^{+5}$	$1.68 \times 10^{+4}$	$1.88 \times 10^{+5}$	$\mathbf{4.41 \times 10^{+3}}$	$6.83 \times 10^{+6}$	$2.20 \times 10^{+7}$	$2.03 \times 10^{+7}$
Ranking first	20	0	1	0	1	3	0	2	3	3	0

Table 9. The mean values for the DTSA, EAT-TSA, fb_TSA, TSA, STSA, MTSA, PSO, GWO, BOA and RSA in 100 dimensions.

Function	DTSA	TSA	STSA	MTSA	EST-TSA	fb-TSA	PSO	GWO	BOA	RSA	GA
Unimodal functions	$\mathbf{5.87 \times 10^{+7}}$	$2.42 \times 10^{+9}$	$1.17 \times 10^{+10}$	$1.34 \times 10^{+8}$	$1.55 \times 10^{+9}$	$2.15 \times 10^{+8}$	$5.60 \times 10^{+8}$	$9.18 \times 10^{+8}$	$1.08 \times 10^{+10}$	$7.88 \times 10^{+9}$	$3.90 \times 10^{+9}$
	$\mathbf{7.81 \times 10^{+4}}$	$5.33 \times 10^{+10}$	$4.67 \times 10^{+11}$	$4.47 \times 10^{+8}$	$6.23 \times 10^{+10}$	$\mathbf{1.02 \times 10^{+9}}$	$1.09 \times 10^{+10}$	$9.34 \times 10^{+10}$	$3.12 \times 10^{+11}$	$2.79 \times 10^{+11}$	$2.16 \times 10^{+11}$
	$\mathbf{2.63 \times 10^{+4}}$	$3.46 \times 10^{+5}$	$8.09 \times 10^{+5}$	$2.14 \times 10^{+5}$	$2.85 \times 10^{+5}$	$3.23 \times 10^{+5}$	$4.46 \times 10^{+5}$	$3.42 \times 10^{+5}$	$3.23 \times 10^{+5}$	$3.08 \times 10^{+5}$	$2.85 \times 10^{+5}$
Simple multimodal functions	$\mathbf{7.81 \times 10^{+2}}$	$9.36 \times 10^{+3}$	$1.59 \times 10^{+5}$	$9.20 \times 10^{+2}$	$1.12 \times 10^{+4}$	$\mathbf{1.05 \times 10^{+3}}$	$2.27 \times 10^{+3}$	$1.11 \times 10^{+4}$	$1.07 \times 10^{+5}$	$8.06 \times 10^{+4}$	$5.35 \times 10^{+4}$
	$\mathbf{5.21 \times 10^{+2}}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$	$5.21 \times 10^{+2}$
	$\mathbf{6.48 \times 10^{+2}}$	$7.42 \times 10^{+2}$	$7.63 \times 10^{+2}$	$6.59 \times 10^{+2}$	$7.38 \times 10^{+2}$	$6.83 \times 10^{+2}$	$7.08 \times 10^{+2}$	$7.08 \times 10^{+2}$	$7.57 \times 10^{+2}$	$7.58 \times 10^{+2}$	$7.50 \times 10^{+2}$
	$\mathbf{7.00 \times 10^{+2}}$	$1.17 \times 10^{+3}$	$4.98 \times 10^{+3}$	$7.06 \times 10^{+2}$	$1.29 \times 10^{+3}$	$7.11 \times 10^{+2}$	$7.85 \times 10^{+2}$	$1.65 \times 10^{+3}$	$3.83 \times 10^{+3}$	$3.47 \times 10^{+3}$	$2.92 \times 10^{+3}$
	$\mathbf{9.55 \times 10^{+2}}$	$1.89 \times 10^{+3}$	$2.57 \times 10^{+3}$	$1.17 \times 10^{+3}$	$1.94 \times 10^{+3}$	$1.19 \times 10^{+3}$	$1.34 \times 10^{+3}$	$1.58 \times 10^{+3}$	$2.15 \times 10^{+3}$	$2.27 \times 10^{+3}$	$2.02 \times 10^{+3}$
	$\mathbf{1.10 \times 10^{+3}}$	$2.11 \times 10^{+3}$	$3.01 \times 10^{+3}$	$1.79 \times 10^{+3}$	$2.21 \times 10^{+3}$	$1.52 \times 10^{+3}$	$1.85 \times 10^{+3}$	$1.70 \times 10^{+3}$	$2.39 \times 10^{+3}$	$2.38 \times 10^{+3}$	$2.26 \times 10^{+3}$
	$\mathbf{7.42 \times 10^{+3}}$	$2.99 \times 10^{+4}$	$3.29 \times 10^{+4}$	$1.71 \times 10^{+4}$	$2.93 \times 10^{+4}$	$1.90 \times 10^{+4}$	$1.77 \times 10^{+4}$	$2.01 \times 10^{+4}$	$3.30 \times 10^{+4}$	$3.07 \times 10^{+4}$	$3.08 \times 10^{+4}$
	$\mathbf{1.61 \times 10^{+4}}$	$3.21 \times 10^{+4}$	$3.28 \times 10^{+4}$	$3.19 \times 10^{+4}$	$2.94 \times 10^{+4}$	$3.20 \times 10^{+4}$	$3.12 \times 10^{+4}$	$2.78 \times 10^{+4}$	$3.26 \times 10^{+4}$	$3.14 \times 10^{+4}$	$3.05 \times 10^{+4}$
	$\mathbf{1.20 \times 10^{+3}}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.20 \times 10^{+3}$	$1.21 \times 10^{+3}$	$1.20 \times 10^{+3}$
	$\mathbf{1.30 \times 10^{+3}}$	$1.30 \times 10^{+3}$	$1.31 \times 10^{+3}$	$1.30 \times 10^{+3}$	$1.30 \times 10^{+3}$	$1.30 \times 10^{+3}$	$1.30 \times 10^{+3}$	$1.31 \times 10^{+3}$	$1.31 \times 10^{+3}$	$1.31 \times 10^{+3}$	$1.31 \times 10^{+3}$
	$\mathbf{1.40 \times 10^{+3}}$	$1.54 \times 10^{+3}$	$2.63 \times 10^{+3}$	$1.40 \times 10^{+3}$	$1.58 \times 10^{+3}$	$1.40 \times 10^{+3}$	$1.43 \times 10^{+3}$	$1.65 \times 10^{+3}$	$2.34 \times 10^{+3}$	$2.22 \times 10^{+3}$	$2.05 \times 10^{+3}$
	$\mathbf{1.55 \times 10^{+3}}$	$6.26 \times 10^{+5}$	$2.15 \times 10^{+8}$	$1.61 \times 10^{+3}$	$4.66 \times 10^{+5}$	$1.77 \times 10^{+3}$	$1.62 \times 10^{+4}$	$2.75 \times 10^{+5}$	$3.05 \times 10^{+7}$	$1.34 \times 10^{+7}$	$4.49 \times 10^{+6}$
	$\mathbf{1.65 \times 10^{+3}}$	$1.65 \times 10^{+3}$	$1.65 \times 10^{+3}$	$1.65 \times 10^{+3}$	$1.65 \times 10^{+3}$	$1.65 \times 10^{+3}$	$1.65 \times 10^{+3}$	$1.65 \times 10^{+3}$	$1.65 \times 10^{+3}$	$1.65 \times 10^{+3}$	$1.65 \times 10^{+3}$

Table 9. *Cont.*

Function	DTSA	TSA	STSA	MTSA	EST-TSA	fb-TSA	PSO	GWO	BOA	RSA	GA
Hybrid functions	**5.02** × **10**$^{+6}$	2.22 × 10^{+8}	1.15 × 10^{+9}	1.18 × 10^{+7}	1.68 × 10^{+8}	2.55 × 10^{+7}	8.84 × 10^{+7}	1.07 × 10^{+8}	2.05 × 10^{+9}	1.27 × 10^{+9}	7.33 × 10^{+8}
	2.45 × **10**$^{+3}$	2.98 × 10^{+3}	2.16 × 10^{+10}	4.69 × 10^{+4}	**5.14** × **10**$^{+4}$	2.99 × 10^{+3}	2.09 × 10^{+7}	2.35 × 10^{+9}	4.47 × 10^{+10}	3.22 × 10^{+10}	2.16 × 10^{+10}
	2.00 × **10**$^{+3}$	2.15 × 10^{+3}	6.32 × 10^{+3}	2.01 × 10^{+3}	2.24 × 10^{+3}	**2.03** × **10**$^{+3}$	2.15 × 10^{+3}	2.57 × 10^{+3}	1.29 × 10^{+4}	8.82 × 10^{+3}	5.74 × 10^{+3}
	5.05 × **10**$^{+4}$	2.70 × 10^{+5}	4.24 × 10^{+6}	1.14 × 10^{+5}	2.35 × 10^{+5}	1.89 × 10^{+5}	2.96 × 10^{+5}	2.89 × 10^{+5}	1.40 × 10^{+6}	8.38 × 10^{+5}	4.30 × 10^{+5}
	3.33 × **10**$^{+6}$	8.85 × 10^{+7}	5.02 × 10^{+8}	6.83 × 10^{+6}	5.72 × 10^{+7}	9.40 × 10^{+6}	3.75 × 10^{+7}	4.44 × 10^{+7}	7.12 × 10^{+8}	3.97 × 10^{+8}	1.83 × 10^{+8}
	4.22 × **10**$^{+3}$	7.12 × 10^{+3}	1.07 × 10^{+4}	6.66 × 10^{+3}	6.75 × 10^{+3}	6.68 × 10^{+3}	6.24 × 10^{+3}	5.96 × 10^{+3}	4.27 × 10^{+5}	1.45 × 10^{+5}	1.88 × 10^{+4}
	2.65 × 10^{+3}	2.77 × 10^{+3}	5.89 × 10^{+3}	2.66 × 10^{+3}	**2.50** × **10**$^{+3}$	2.66 × 10^{+3}	2.72 × 10^{+3}	3.12 × 10^{+3}	**2.50** × **10**$^{+3}$	**2.50** × **10**$^{+3}$	3.14 × 10^{+3}
Composition functions	2.79 × 10^{+3}	3.00 × 10^{+3}	3.93 × 10^{+3}	2.78 × 10^{+3}	2.60 × 10^{+3}	2.82 × 10^{+3}	2.92 × 10^{+3}	2.60 × 10^{+3}	2.60 × 10^{+3}	**2.60** × **10**$^{+3}$	2.75 × 10^{+3}
	2.78 × 10^{+3}	3.06 × 10^{+3}	3.85 × 10^{+3}	2.77 × 10^{+3}	**2.70** × **10**$^{+3}$	2.81 × 10^{+3}	2.86 × 10^{+3}	2.74 × 10^{+3}	**2.70** × **10**$^{+3}$	**2.70** × **10**$^{+3}$	2.73 × 10^{+3}
	2.80 × **10**$^{+3}$	2.99 × 10^{+3}	**2.72** × **10**$^{+3}$	2.80 × 10^{+3}	2.80 × 10^{+3}	2.81 × 10^{+3}	2.85 × 10^{+3}	2.86 × 10^{+3}	2.80 × 10^{+3}	2.80 × 10^{+3}	2.80 × 10^{+3}
	4.02 × **10**$^{+3}$	6.42 × 10^{+3}	7.46 × 10^{+3}	4.45 × 10^{+3}	6.45 × 10^{+3}	4.93 × 10^{+3}	5.79 × 10^{+3}	5.94 × 10^{+3}	7.99 × 10^{+3}	7.57 × 10^{+3}	8.41 × 10^{+3}
	6.49 × 10^{+3}	2.13 × 10^{+4}	2.30 × 10^{+4}	8.19 × 10^{+3}	2.18 × 10^{+4}	1.02 × 10^{+4}	1.43 × 10^{+4}	**5.37** × **10**$^{+3}$	3.00 × 10^{+4}	2.29 × 10^{+4}	3.56 × 10^{+4}
	5.73 × 10^{+3}	2.50 × 10^{+7}	1.43 × 10^{+9}	1.63 × 10^{+5}	4.53 × 10^{+7}	2.73 × 10^{+4}	4.56 × 10^{+6}	**3.14** × **10**$^{+3}$	**3.10** × **10**$^{+3}$	**3.10** × **10**$^{+3}$	2.13 × 10^{+9}
	5.56 × 10^{+4}	2.65 × 10^{+6}	5.64 × 10^{+7}	1.46 × 10^{+5}	4.23 × 10^{+6}	1.97 × 10^{+5}	8.55 × 10^{+5}	**6.14** × **10**$^{+3}$	3.36 × 10^{+7}	3.63 × 10^{+7}	1.28 × 10^{+8}
Ranking first	23	0	1	0	2	0	0	2	3	4	0

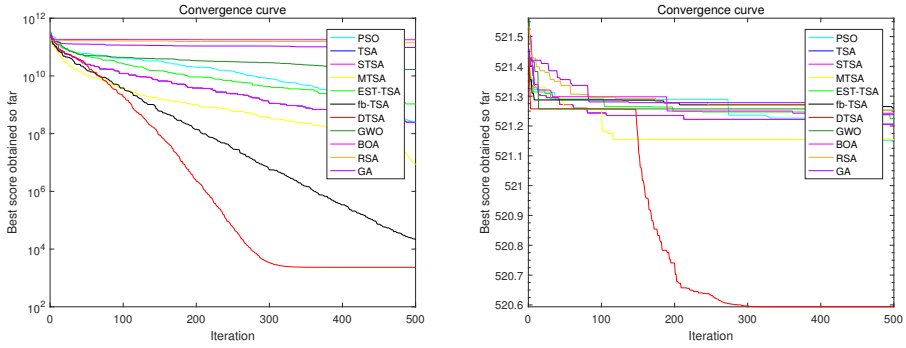


(**a**) f_3, unimodal function (**b**) f_6, multimodal function

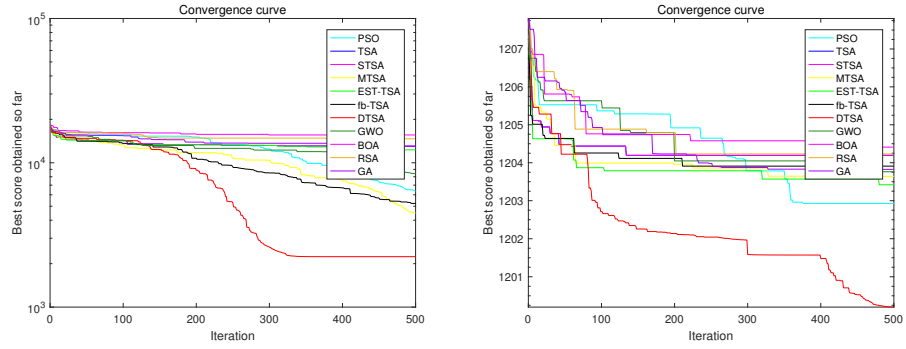
(**c**) f_{10}, multimodal function (**d**) f_{17}, multimodal function

(**e**) f_{24}, hybrid function (**f**) f_{25}, composite function

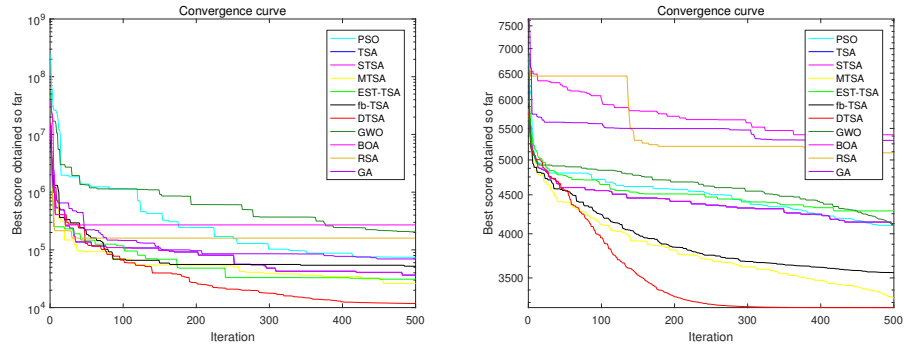
Figure 18. Convergence curves of the DTSA and TSA and its recent variants in 30 dimensions.



(**a**) f_3, unimodal function (**b**) f_6, multimodal function

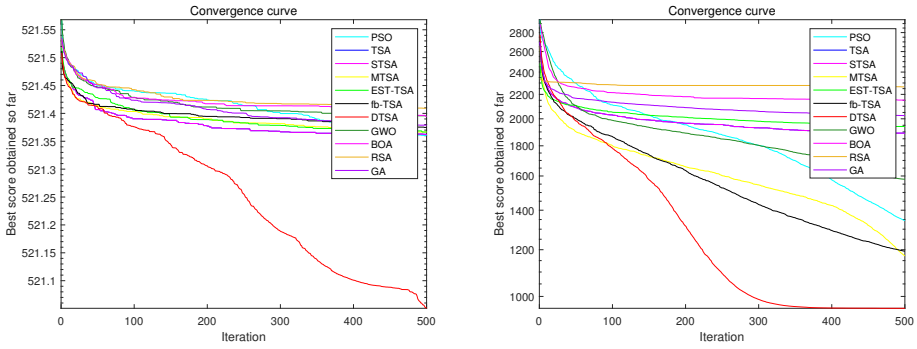


(**c**) f_{10}, multimodal function (**d**) f_{17}, multimodal function



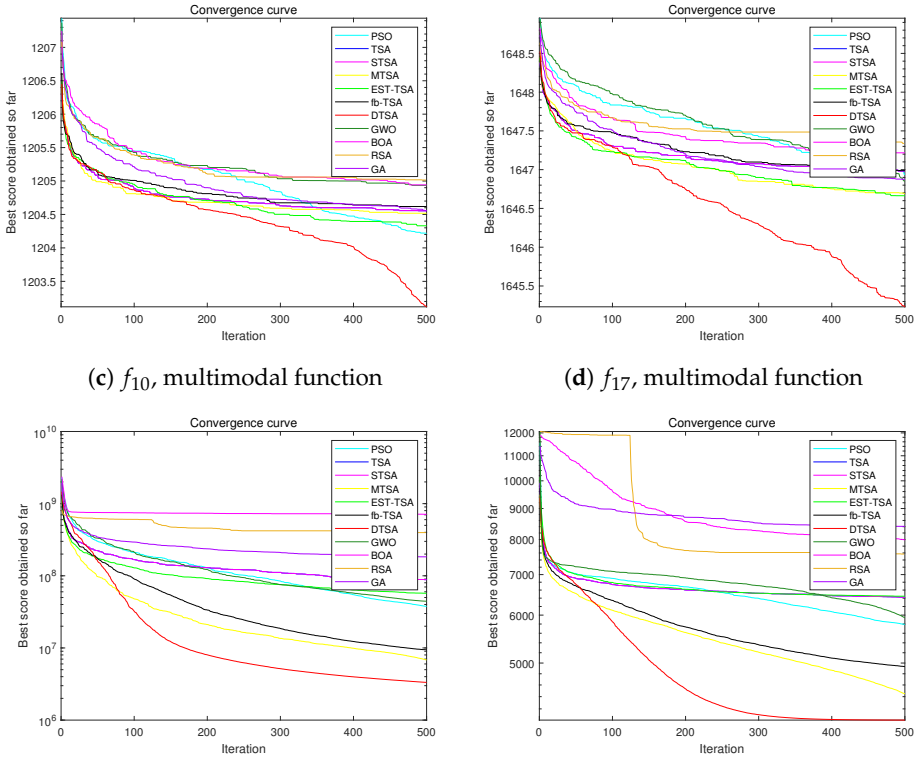
(**e**) f_{24}, hybrid function (**f**) f_{25}, composite function

Figure 19. Convergence curve of the DTSA and TSA and its recent variants in 50 dimensions.



(**a**) f_3, unimodal function (**b**) f_6, multimodal function

Figure 20. *Cont.*



(**c**) f_{10}, multimodal function (**d**) f_{17}, multimodal function

(**e**) f_{24}, hybrid function (**f**) f_{25}, composite function

Figure 20. Convergence curve of the DTSA and TSA and its recent variants in 100 dimensions.

4.3.3. Comparative Experiment 3: Analyzing the Stability of the DTSA

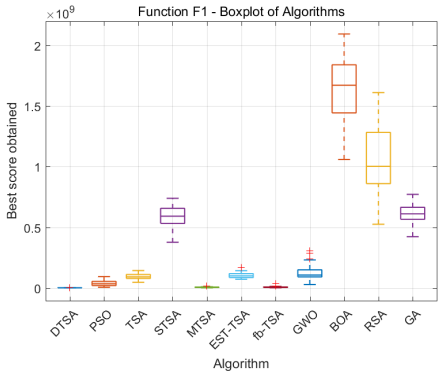
In this section, boxplots are presented to facilitate stability analysis of the DTSA algorithm. Figures 21–23 depict these boxplots, with the x-axis representing the algorithm proposed in this study and other comparative algorithms. The y-axis shows objective function evaluations derived from 30 rounds of experiments for each algorithm.

Each plot displays uppermost and lowermost black lines representing maximum and minimum values from the 30 evaluations. The bounds of the box signify upper and lower quartiles, while the red line within the box indicates the median. Plus signs outside the box highlight outliers, denoting exceptional or subpar performance instances in specific rounds. This graphical representation aids in comprehensive assessment of the algorithm's stability and performance across multiple experimental rounds.

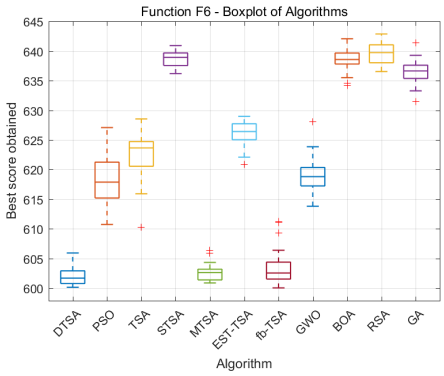
Upon scrutinizing the comparative boxplots, it is evident that DTSA demonstrates commendable consistency in performance across various experimental setups. Its robust boxplot manifestations and minimal presence of outliers substantiate DTSA's capability to consistently yield favorable results in diverse experimental environments, affirming its distinguished attribute of elevated stability. This empirical validation underscores DTSA's reliability and effectiveness in optimization tasks, confirming its utility in complex problem-solving scenarios with confidence.

In summary, DTSA's performance in the quantitative analysis highlights a substantial boost in its optimization efficacy. By integrating a PSO-inspired seed generation mechanism and a count-based adaptive strategy, it effectively addresses the traditional TSA's limitations regarding premature convergence and challenges with high-dimensional, intricate optimization issues. Notably, the experiments not only validate DTSA's superiority through comparisons with numerous cutting-edge algorithms but also reinforce its efficiency and stability in real-world problem applications. The comprehensive assessment on the IEEE CEC 2014 benchmark functions demonstrates DTSA's rapid and steady convergence trend, particularly excelling in navigating multimodal, mixed, and composite functions by adeptly avoiding local optima to reach global optimality.

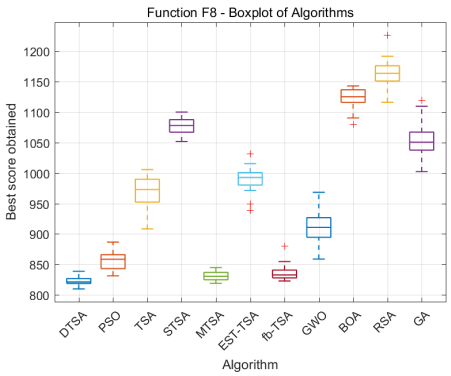
However, when confronted with simplistic, unimodal functions, where all algorithms effortlessly locate the optimal solution due to the inherently low complexity, the enhancements brought by DTSA become less conspicuous. In alignment with the no free lunch theorem [44], it is a recognized fact that no single algorithm can universally prevail in every scenario; their superiority is inherently tied to the specific complexities embedded within each distinct problem. Hence, it is inevitable that DTSA exhibits a differentiated effectiveness, finely attuned to the intricacy and dimensional breadth of the tasks it encounters.



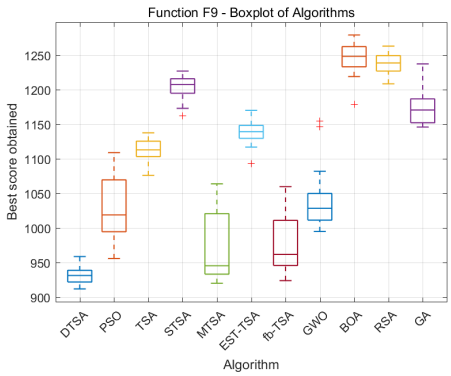
(**a**) f_1, unimodal function



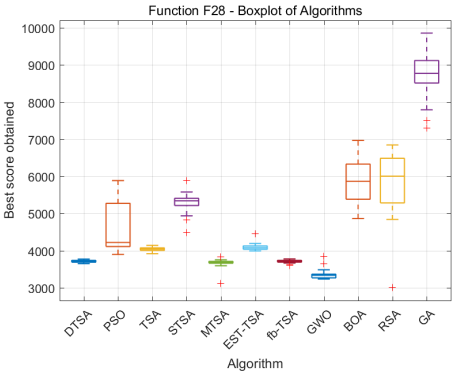
(**b**) f_6, multimodal function



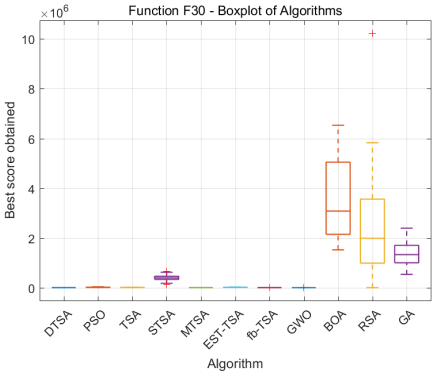
(**c**) f_8, multimodal function



(**d**) f_9, multimodal function

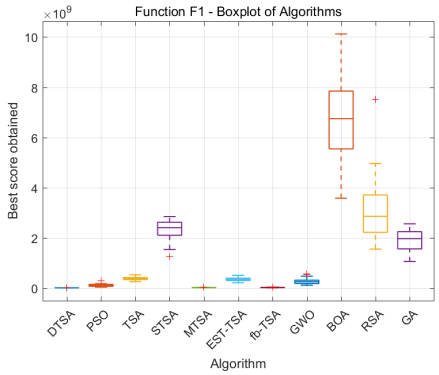


(**e**) f_{28}, composite function

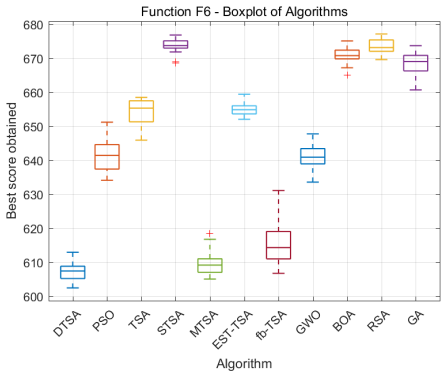


(**f**) f_{30}, composite function

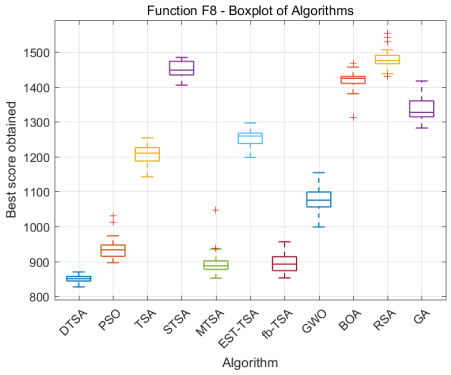
Figure 21. Boxplots of all experiment algorithms in 30 dimensions.



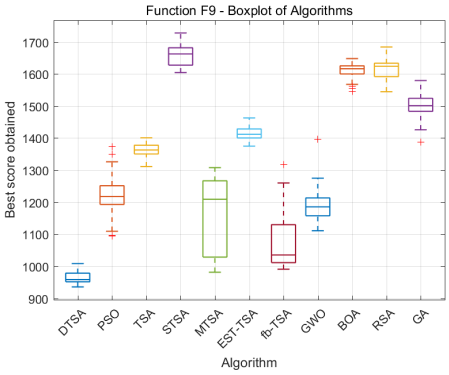
(**a**) f_1, unimodal function



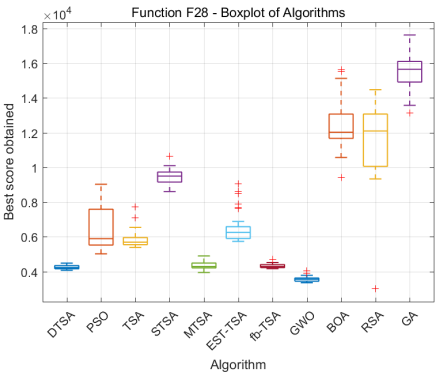
(**b**) f_6, multimodal function



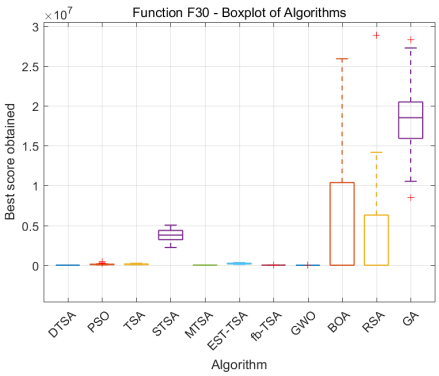
(**c**) f_8, multimodal function



(**d**) f_9, multimodal function

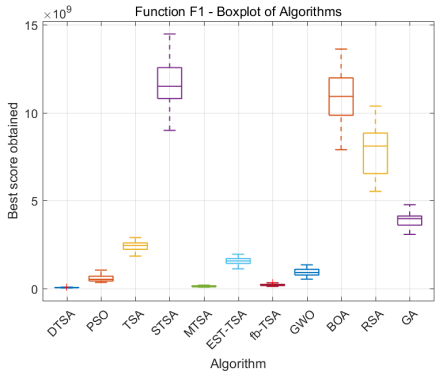


(**e**) f_{28}, composite function

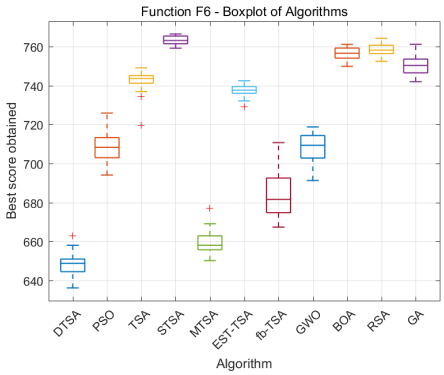


(**f**) f_{30}, composite function

Figure 22. Boxplots of all experiment algorithms in 50 dimensions.

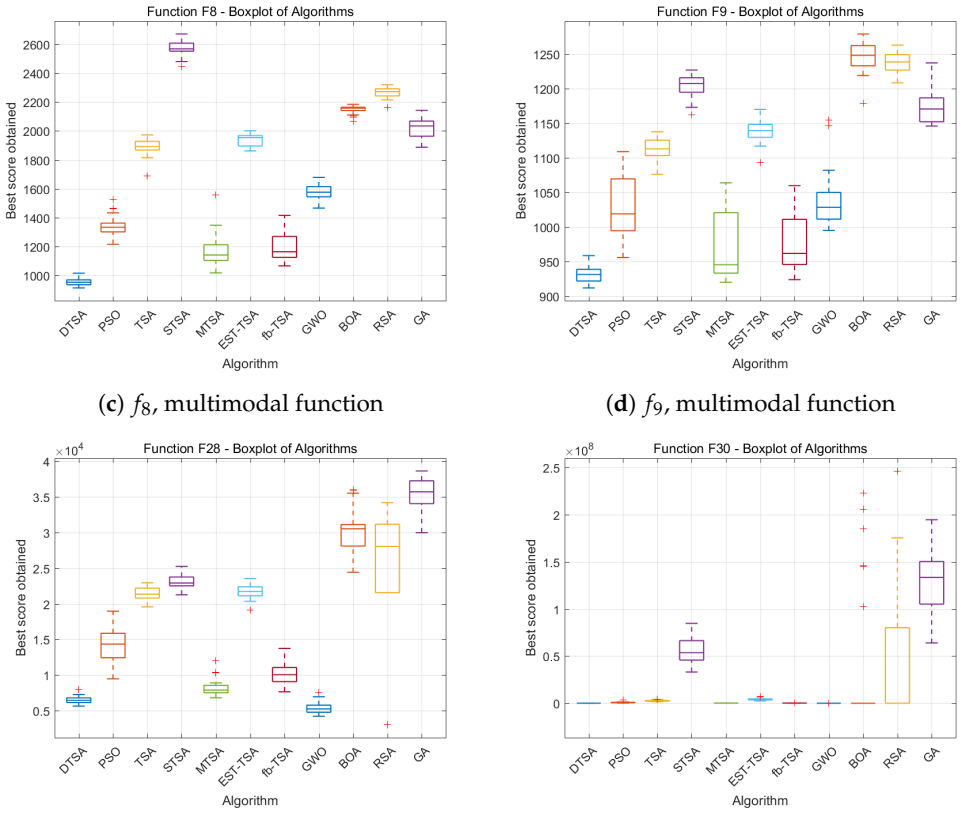


(**a**) f_1, unimodal function



(**b**) f_6, multimodal function

Figure 23. *Cont.*



(**c**) f_8, multimodal function (**d**) f_9, multimodal function

(**e**) f_{28}, composite function (**f**) f_{30}, composite function

Figure 23. Boxplots of all experiment algorithms in 100 dimensions. .

4.4. Further Analysis

To elucidate the performance characteristics of DTSA, it is imperative to undertake an advanced analysis that integrates findings from both qualitative and quantitative investigations. This approach will provide a comprehensive understanding of the experimental observations associated with DTSA's functionality.

Firstly, DTSA showcases superior performance over several TSA variants, as it integrates three key innovative mechanisms that collectively enhance its operational efficiency. These innovations significantly improve DTSA's adaptability, diversity, and convergence speed. By dynamically updating seed positions, adjusting search behavior to prevent premature convergence, and incorporating genetic diversity through arithmetic crossover and natural selection, DTSA achieves a remarkable balance between exploration and exploitation. This strategic combination not only accelerates the algorithm's convergence towards optimal solutions but also maintains a high level of population diversity, setting DTSA apart in optimization tasks and showcasing its advanced capability in handling complex optimization problems

Secondly, in addressing low-dimensional optimization challenges, DTSA does not exhibit a marked superiority over competing algorithms, attributable to its sophisticated mechanisms primarily optimized for the intricacies of high-dimensional search spaces. These mechanisms, while potent in navigating and exploiting the complex landscapes of high-dimensional problems, yield diminishing returns in less complex, low-dimensional scenarios. Consequently, the inherent advantage of DTSA's advanced features becomes less pronounced, as simpler algorithmic solutions prove equally adept in these contexts. This observation underscores a critical avenue for future research aimed at refining DTSA's adaptability and efficiency across a diverse array of problem scales, thereby enhancing its utility in a wider spectrum of optimization tasks.

Thirdly, the empirical evidence, as delineated by the boxplot visualizations in Figures 21–23, unequivocally establishes the superior stability and quality of DTSA in comparison to the

algorithms it was tested against within this study. Specifically, the more compact and lower interquartile ranges of DTSA's boxplots for certain functions distinctly underscore its enhanced performance metrics. This excellence is directly attributable to the innovative integration of the tree population integrated evolutionary strategy and adaptive velocity-driven seed generation mechanisms, as introduced in this research. These methodologies empower the DTSA to dynamically refine the optimization trajectory across varied benchmark function evaluations, culminating in reduced variability (as evidenced by smaller standard deviations) and improved reliability in experimental outcomes. This adaptability and precision in handling diverse optimization scenarios underscore the algorithm's robustness and its potential applicability in solving complex optimization problems with heightened efficiency and consistency.

4.5. Statistical Experiments

Table 10 presents the outcomes of the Wilcoxon's signed-rank test applied to the experimental data of the DTSA algorithm in comparison to 11 other algorithms [66]. The table includes p-values obtained at the significance level α. Columns labeled TRUE and FALSE indicate whether the hypothesis is rejected or not rejected at the specified significance level of α. The data for this analysis are sourced from the comparative experiments detailed in Section 4.3, ensuring consistency with the experimental setup.

Our assertion is that DTSA demonstrates superior performance compared to the other algorithms. A smaller p-value resulting from the experiment between DTSA and a comparative algorithm, below the significance level α, indicates DTSA's superior performance. Tables 10 reveals that DTSA consistently exhibits very low *p*-values compared to the majority of the algorithms. Consequently, under both significance levels (α = 0.1 and α = 0.05), our hypothesis is confirmed. These findings affirm that, in this experiment, the DTSA algorithm significantly outperforms the other algorithms in terms of performance.

Table 10. Results of Wilcoxon's test for DTSA and other algorithms.

$\alpha = 0.1$	Algorithms									
	TSA	**STSA**	**MTSA**	**EST-TSA**	**fb-TSA**	**PSO**	**GWO**	**BOA**	**RSA**	**GA**
D = 30	1.13×10^{-5} **TRUE**	3.18×10^{-6} **TRUE**	4.68×10^{-3} **TRUE**	1.60×10^{-4} **TRUE**	1.29×10^{-3} **TRUE**	1.73×10^{-6} **TRUE**	8.31×10^{-4} **TRUE**	1.24×10^{-5} **TRUE**	1.24×10^{-5} **TRUE**	3.52×10^{-6} **TRUE**
D = 50	4.29×10^{-6} **TRUE**	3.18×10^{-6} **TRUE**	1.36×10^{-5} **TRUE**	4.45×10^{-5} **TRUE**	2.11×10^{-3} **TRUE**	1.73×10^{-6} **TRUE**	5.71×10^{-4} **TRUE**	5.31×10^{-5} **TRUE**	4.86×10^{-5} **TRUE**	6.34×10^{-6} **TRUE**
D = 100	1.73×10^{-6} **TRUE**	2.88×10^{-6} **TRUE**	1.2×10^{-5} **TRUE**	2.60×10^{-5} **TRUE**	1.73×10^{-6} **TRUE**	1.73×10^{-6} **TRUE**	7.16×10^{-4} **TRUE**	6.32×10^{-5} **TRUE**	6.32×10^{-5} **TRUE**	6.34×10^{-6} **TRUE**
$\alpha = 0.5$	Algorithms									
D = 30	1.13×10^{-5} **TRUE**	3.18×10^{-6} **TRUE**	4.68×10^{-3} **TRUE**	1.60×10^{-4} **TRUE**	1.29×10^{-3} **TRUE**	1.73×10^{-6} **TRUE**	8.31×10^{-4} **TRUE**	1.24×10^{-5} **TRUE**	1.24×10^{-5} **TRUE**	3.52×10^{-6} **TRUE**
D = 50	4.29×10^{-6} **TRUE**	3.18×10^{-6} **TRUE**	1.36×10^{-5} **TRUE**	4.45×10^{-5} **TRUE**	2.11×10^{-3} **TRUE**	1.73×10^{-6} **TRUE**	5.71×10^{-4} **TRUE**	5.31×10^{-5} **TRUE**	4.86×10^{-5} **TRUE**	6.34×10^{-6} **TRUE**
D = 100	1.73×10^{-6} **TRUE**	2.88×10^{-6} **TRUE**	1.2×10^{-5} **TRUE**	2.60×10^{-5} **TRUE**	1.73×10^{-6} **TRUE**	1.73×10^{-6} **TRUE**	7.16×10^{-4} **TRUE**	6.32×10^{-5} **TRUE**	6.32×10^{-5} **TRUE**	6.34×10^{-6} **TRUE**

4.6. Practical Engineering Problems of Mathematical Modeling

In light of the advanced analysis delineated in Section 4.4, which evidences the Dynamic Tree-Seed Algorithm's (DTSA) proficiency in navigating high-dimensional complex problem spaces, this section is dedicated to an empirical evaluation of DTSA's adeptness in addressing complex constrained optimization challenges. Specifically, we scrutinize its performance through the lens of intricate objective functions and multifarious constraints, employing a suite of benchmark problems for this purpose: Tension spring design [67], three-bar truss design [68], welded beam design [69], cantilever beam design [70], and step-cone pulley design problems [71]. The analytical results, encapsulated in Tables 11–15, are articulated through several statistical measures: best signifies the optimal value achieved by DTSA across 30 experimental iterations; mean delineates the average result; std refers to the standard deviation, indicating result variability; worst captures the least favorable outcome; and the x denotes the optimal solution configuration ascertained by DTSA at the juncture of achieving its best result. This methodological approach affords a rigorous

assessment of DTSA's capability to effectively resolve complex constrained optimization tasks, furnishing a comprehensive perspective on its performance efficacy.

4.6.1. Example 1: Tension Spring Design Problem

The pressure vessel design (PVD) problem aims to minimize the total cost, denoted as $f(x)$, while satisfying the production requirements. This optimization problem is defined by four design variables: the shell thickness $T_s(x_3)$, the head thickness $T_h(x_4)$, the inner radius $R(x_1)$, and the length of the vessel $L(x_2)$, excluding the head. Both T_s and T_h are quantized in multiples of 0.625, whereas R and L are considered as continuous variables. The mathematical model of the pressure vessel design (PVD) problem is detailed from [67]:

Objective function:

$$\text{minimize } f(x) = (N+2)Dd^2$$

Subject to constraints:

$$g_1(x) = 1 - \frac{D^3 N}{71785d^4} \leq 0$$

$$g_2(x) = \frac{4D^2 - dD}{12566(Dd^3 - d^4)} + \frac{1}{5108d^2} - 1 \leq 0$$

$$g_3(x) = 1 - \frac{140.45d}{D^2 N} \leq 0$$

$$g_4(x) = \frac{D+d}{1.5} - 1 \leq 0$$

Variable bounds:

$$0.05 \leq x_1 \leq 2, \quad 0.25 \leq x_2 \leq 1.3, \quad 2 \leq x_3 \leq 15$$

The results of the tension spring design problem are shown in Table 11. The results show that the DTSA has better performance.

Table 11. Tension spring design problem.

	DTSA	TSA	DBO	HHO	GWO	SO	WFO	GOA
Best	**0.013**	0.013	0.013	0.014	0.013	0.013	0.018	0.013
Mean	**0.013**	0.013	0.013	0.015	0.013	0.013	0.036	0.014
Std	0.000	0.000	0.000	0.001	0.000	0.000	0.026	0.001
Worst	0.000	0.013	0.013	0.016	0.013	0.013	0.054	0.015
X_1	0.053	0.053	0.050	0.060	0.050	0.050	0.068	0.058
X_2	0.389	0.381	0.317	0.597	0.319	0.317	0.872	0.521
X_3	9.602	10.015	14.028	4.426	13.918	14.028	2.439	5.648

4.6.2. Example 2: Three-Bar Truss Design Problem

The efficacy of the algorithm is evaluated in test 2, utilizing the three-bar truss design problem. The primary objective is to minimize the volumetric attribute of the statically loaded three-bar truss, constrained by a triplet of inequality conditions. The optimization's objective function is structured to compute the optimal cross-sectional areas corresponding to the design variables, denoted as x_1 and x_2. This delineation is emblematic of structural optimization paradigms, where the optimization of material usage and adherence to mechanical constraints are of critical importance, thus providing a stringent assessment of the algorithm's navigational acumen within multifaceted design domains.

Consider $\mathbf{X} = [x_1, x_2]$,

Objective function:

$$\text{minimize } f(\mathbf{X}) = (2\sqrt{2}x_1 + x_2) \times l$$

Subject to constraints:

$$g_1(\mathbf{X}) = \frac{\sqrt{2x_1} + x_2}{\sqrt{2x_1^2 + 2x_1x_2}} - P - \sigma \leq 0,$$
$$g_2(\mathbf{X}) = \frac{x_2}{\sqrt{2x_1^2 + 2x_1x_2}} - P - \sigma \leq 0,$$
$$g_3(\mathbf{X}) = \frac{1}{\sqrt{2x_2 + x_1}} - P - \sigma \leq 0,$$

Where:

$$l = 100 \text{ cm}, \ P = 2 \text{ kN/cm}^3, \ \sigma = 2 \text{ kN/cm}^3,$$

Variable bounds:

$$0 \leq x_1, x_2 \leq 1.$$

The results of the three-bar truss design problem are shown in Table 12. The results show that the DTSA has better performance.

Table 12. Three-bar truss design problem.

	DTSA	**TSA**	**DBO**	**HHO**	**GWO**	**SO**	**WFO**	**GOA**
Best	**263.8958**	$2.64 \times 10^{+2}$	263.8963	263.9051	263.8982	263.8959	265.4799	263.9054
Mean	**263.8958**	263.8958	263.8968	264.0205	263.9012	263.8969	265.5999	263.975
Std	2.81×10^{-6}	3.91×10^{-6}	7.95×10^{-4}	0.163226	4.28×10^{-3}	0.001494	0.169762	0.098407
Worst	$2.64 \times 10^{+2}$	263.8959	263.8974	264.1359	263.9043	263.898	265.4799	264.0446
X_1	0.7886	0.7887	0.7879	0.7851	0.7895	0.7885	0.822665	0.785091
X_2	0.4082	0.4081	0.4104	0.4182	0.4059	0.4087	0.3279	0.4184

4.6.3. Example 3: Welded Beam Design Problem

The welded beam design (WBD) problem is a classical optimization task that seeks to minimize the manufacturing cost of a beam's design. The problem is defined by four design variables: the beam's length l, height t, thickness b, and the weld thickness h. The objective of the optimization process is to ascertain the values of these variables that result in the minimum production cost while adhering to the constraints imposed by shear stress τ, bending stress θ, buckling load on the bar P_c, end deflection δ, and other prescribed boundary conditions. The WBD problem thus serves as a paradigmatic nonlinear programming challenge, wherein the cost function is intricately linked with a set of physical constraints that are interdependent. The mathematical formulation of the WBD problem is as follows:

Variables:

$$\vec{l} = [l_1, l_2, l_3, l_4] = [h, l, t, b] = [x_1, x_2, x_3, x_4]$$

Objective function:

$$\text{minimize } f(\vec{l}) = 1.10471 l_1^2 l_2 + 0.04811 l_3 l_4 (14.0 + l_2)$$

Variable bounds:

$$0.1 \leq l_1 \leq 2$$
$$0.1 \leq l_2 \leq 10$$
$$0.1 \leq l_3 \leq 10$$
$$0.1 \leq l_4 \leq 2$$

Subject to constraints:

$$s_1(\vec{l}) = \tau(\vec{l}) - \tau_{\max} \leq 0$$
$$s_2(\vec{l}) = \sigma(\vec{l}) - \sigma_{\max} \leq 0$$
$$s_3(\vec{l}) = \delta(\vec{l}) - \delta_{\max} \leq 0$$
$$s_4(\vec{l}) = l_1 - l_4 \leq 0$$
$$s_5(\vec{l}) = P - P_c(\vec{l}) \leq 0$$
$$s_6(\vec{l}) = 0.125 - l_1 \leq 0$$
$$s_7(\vec{l}) = 1.10471 l_1^2 + 0.04811 l_3 l_4 (14.0 + l_2) - 5.0 \leq 0$$

where:

$\sigma_{\max} = 30,000$ psi, $P = 6000$ lb, $L = 14$ in., $\delta_{\max} = 0.25$ in., $E = 3 \times 10^6$ psi, $\tau_{\max} = 136,000$ psi, $G = 1.2 \times 10^7$ psi.

$$\tau(\vec{l}) = \sqrt{\tau'^2 + 2\tau'\tau''\left(\frac{l_2}{R}\right) + (\tau'')^2}$$
$$\tau' = \frac{P}{\sqrt{2} l_1 l_2}, \quad \tau'' = \frac{MR}{J}, \quad M = p\left(L + \frac{l_2}{2}\right)$$
$$R = \sqrt{\frac{l_2^2 + (l_1 + l_3)^2}{4}}$$
$$J = 2\left\{\sqrt{2} l_1 l_2 \left[\frac{l_2^2}{12} + \frac{(l_1 + l_3)^2}{14}\right]\right\}$$
$$P_c(\vec{l}) = \frac{4.013 E l_3 l_4^2}{6L^2}\left(1 - \frac{l_3\sqrt{E}}{8LG}\right)$$

The results of the welded beam design problem are shown in Table 13. The results show that the DTSA has better performance.

Table 13. Welded beam design problem.

	DTSA	**TSA**	**DBO**	**HHO**	**GWO**	**SO**	**WFO**	**GOA**
Best	**1.6927**	1.6927	1.6927	1.7399	1.6954	1.6928	1.9850	2.3256
Mean	**1.6927**	1.6927	1.6927	1.7838	1.6957	1.6940	2.0007	2.6348
Std	0	1.03×10^{-11}	5.55×10^{-6}	0.0620	0.0004	0.001	0.02226	0.4372
Worst	1.6927	1.6927	1.6927	1.8276	1.6960	1.6951	2.0164	2.9439
X_1	0.2057	0.2057	0.2057	0.1877	0.2055	0.2057	0.24336	0.278417
X_2	3.2349	3.2349	3.2349	3.5996	3.2416	3.2346	2.8060	3.1226
X_3	9.0366	9.0366	9.0366	9.2250	9.0506	9.0372	8.5761	6.7455
X_4	0.2057	0.2057	0.2057	0.2048	0.2056	0.2057	0.2597	0.3704

4.6.4. Example 4: Cantilever Beam Design Problem

The cantilever beam optimization problem represents a fundamental structural engineering design challenge focused on the weight reduction of a cantilever beam featuring a square cross-section. This beam is characterized by a fixed support at one end and experiences a vertical load at the free end. Comprised of five hollow square segments of uniform thickness—specifically, 2/3 of an inch—the primary design variables are the heights or

widths of these segments. The optimization seeks to minimize the total weight while ensuring compliance with structural requirements, including stress limits, deflection tolerances, and vibrational characteristics. It also considers practical aspects such as manufacturability and cost-effectiveness. The mathematical model encapsulating this optimization problem involves an objective function for weight calculation and a set of constraints ensuring the beam's structural feasibility and functional adequacy under applied loads. This problem can be represented by the following mathematical equation:

Objective function:

$$f(\mathbf{X}) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5)$$

Subject to constraints:

$$g(\mathbf{X}) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \leq 0$$

Variable bounds:

$$0.01 \leq x_i \leq 100, \quad i = 1, 2, \ldots, 5$$

The results of the cantilever beam design problem are shown in Table 14. The results show that the DTSA has better performance.

Table 14. Cantilever beam design problem.

	DTSA	TSA	DBO	HHO	GWO	SO	WFO	GOA
Best	**1.3399**	1.3399	1.3399	1.3437	1.3400	1.3400	2.6214	1.3737
Mean	**1.3399**	1.3399	1.3399	1.3440	1.3400	1.3400	2.6740	1.4063
Std	2.08×10^{-7}	7.40×10^{-7}	6.63×10^{-6}	0.0004	5.03×10^{-5}	2.11×10^{-5}	0.0743	0.0462
Worst	1.3399	1.3399	1.3399	1.3443	1.3401	1.3400	2.7266	1.4390
X_1	6.0161	6.0192	6.0354	5.8279	6.0266	5.9732	4.9741	5.9102
X_2	5.3094	5.3085	5.3191	5.2299	5.2760	5.3042	16.3215	6.0045
X_3	4.4947	4.4934	4.4922	4.8202	4.4834	4.5395	7.1959	3.9133
X_4	3.5007	3.4994	3.4801	3.4528	3.5322	3.5006	10.6454	4.1481
X_5	2.1525	2.1529	2.1471	2.2033	2.1565	2.1575	2.8738	2.0382

4.6.5. Example 5: Step-Cone Pulley Problem

The step-cone pulley design problem is a weight minimization task encountered in structural engineering, focusing on the optimization of a multi-stepped pulley system. Each step of the pulley, with a distinct diameter, contributes to the overall design that must adhere to specific mechanical constraints and performance goals. The challenge lies in determining the optimal configuration of these diameters and other design aspects to achieve minimal weight without compromising the pulley's functional integrity. This problem is emblematic of complex engineering design tasks, which require sophisticated optimization techniques to solve due to their nonlinear and constrained nature.

Objective function:

$$\text{minimize } f(\mathbf{x}) = \rho\omega\left[d_1^2\left(1 + \left(\frac{N_1}{N}\right)^2\right) + d_2^2\left(1 + \left(\frac{N_2}{N}\right)^2\right) + d_3^2\left(1 + \left(\frac{N_3}{N}\right)^2\right) + d_4^2\left(1 + \left(\frac{N_4}{N}\right)^2\right)\right]$$

Subject to constraints:

$$\begin{aligned} h_1(\mathbf{x}) &= C_1 - C_2 = 0, \\ h_2(\mathbf{x}) &= C_1 - C_3 = 0, \\ h_3(\mathbf{x}) &= C_1 - C_4 = 0, \\ g_{1,2,3,4}(\mathbf{x}) &\geq R_i \geq 2, \\ g_{5,6,7,8}(\mathbf{x}) &= P_i \geq (0.75 \times 745.6998), \end{aligned}$$

Where:

$$C_i = \frac{\pi d_i}{2}\left(1 + \frac{N_i}{N} + \left(\frac{N_i}{N} - 1\right)^2 \frac{1}{4a} + 2a\right) \quad i = (1,2,3,4).$$

$$R_i = \exp\left[\mu\left(\pi - 2\sin^{-1}\left(\frac{(N_i/N) - 1}{2a}\right)\right)\right] \quad i = (1,2,3,4).$$

$$P_i = \rho\omega\left[1 - \exp\left(-\mu\left[\pi - 2\sin^{-1}\left(\frac{(N_i/N) - 1}{2a}\right)\right]\right)\right]\frac{\pi d_i N_i}{60} \quad i = (1,2,3,4).$$

$$\rho = 7200\ \text{kg/m}^3,\ a = 3\ \text{m},\ \mu = 0.35,\ s = 1.75\ \text{MPa},\ t = 8\ \text{mm}$$

The results of the step-cone pulley problem are shown in Table 15. The results show that the DTSA has better performance.

Table 15. Step-cone pulley problem.

	DTSA	TSA	DBO	HHO	GWO	SO	WFO	GOA
Best	$1.67 \times 10^{+1}$	$2.84 \times 10^{+81}$	$1.67 \times 10^{+1}$	$4.60 \times 10^{+87}$	$4.31 \times 10^{+92}$	$1.67 \times 10^{+1}$	$6.77 \times 10^{+103}$	$4.86 \times 10^{+93}$
Mean	$1.67 \times 10^{+1}$	$2.49 \times 10^{+84}$	$1.75 \times 10^{+1}$	$2.17 \times 10^{+91}$	$8.53 \times 10^{+92}$	$1.44 \times 10^{+75}$	$1.03 \times 10^{+104}$	$3.01 \times 10^{+96}$
Std	1.17×10^{-1}	$3.51 \times 10^{+3}$	$1.08 \times 10^{+0}$	$3.07 \times 10^{+91}$	$5.97 \times 10^{+92}$	$2.03 \times 10^{+75}$	$5.00 \times 10^{+103}$	$4.25 \times 10^{+96}$
Worst	$1.68 \times 10^{+1}$	$4.97 \times 10^{+84}$	$1.83 \times 10^{+1}$	$4.34 \times 10^{+91}$	$1.28 \times 10^{+93}$	$2.87 \times 10^{+75}$	$1.38 \times 10^{+104}$	$6.01 \times 10^{+96}$
X_1	$3.98 \times 10^{+1}$	$3.91 \times 10^{+1}$	$4.00 \times 10^{+1}$	$3.97 \times 10^{+1}$	$3.95 \times 10^{+1}$	$3.89 \times 10^{+1}$	$5.82 \times 10^{+1}$	$3.89 \times 10^{+1}$
X_2	$5.47 \times 10^{+1}$	$5.38 \times 10^{+1}$	$5.50 \times 10^{+1}$	$5.46 \times 10^{+1}$	$5.44 \times 10^{+1}$	$5.35 \times 10^{+1}$	$5.98 \times 10^{+1}$	$5.35 \times 10^{+1}$
X_3	$7.29 \times 10^{+1}$	$7.17 \times 10^{+1}$	$7.33 \times 10^{+1}$	$7.28 \times 10^{+1}$	$7.26 \times 10^{+1}$	$7.14 \times 10^{+1}$	$7.78 \times 10^{+1}$	$7.14 \times 10^{+1}$
X_4	$8.75 \times 10^{+1}$	$8.59 \times 10^{+1}$	$8.79 \times 10^{+1}$	$8.73 \times 10^{+1}$	$8.70 \times 10^{+1}$	$8.56 \times 10^{+1}$	$8.23 \times 10^{+1}$	$8.55 \times 10^{+1}$
X_5	$8.69 \times 10^{+1}$	$8.92 \times 10^{+1}$	$8.65 \times 10^{+1}$	$8.73 \times 10^{+1}$	$8.95 \times 10^{+1}$	$8.99 \times 10^{+1}$	$8.81 \times 10^{+1}$	$8.92 \times 10^{+1}$

5. Conclusions and Future Work

The Dynamic Tree-Seed Algorithm presents a series of quantitative improvements over existing optimization algorithms, which can be specifically highlighted in terms of numerical results and comparative evaluations. In the studies and experiments conducted, DTSA demonstrates its efficacy through the following achievements:

- **Performance enhancement**: DTSA was tested against various benchmarks and recent TSA variants such as STSA, EST-TSA, fb-TSA, and MTSA, along with established algorithms like GA, PSO, GWO, BA, and RSA. It consistently outperformed these algorithms across multiple dimensions (30D, 50D, 100D) and on different types of functions (unimodal, multimodal, composite), as shown in the IEEE CEC 2014 benchmark tests.
- **Convergence and robustness**: The convergence curves depicted in figures like Figure 17 illustrate DTSA's faster convergence rate and stability even in higher dimensional spaces and for complex functions such as hybrid and composite ones. This indicates that DTSA effectively balances exploration and exploitation, leading to quicker and more accurate solutions.
- **Statistical measures**: Across experiments, DTSA's performance was quantified using measures like best, mean, std (standard deviation), worst, and X (optimal solution configuration). These metrics provided a comprehensive view of its effectiveness, showing consistent superiority in finding optimal or near-optimal solutions with reduced variability.
- **Engineering applications**: When applied to real-world engineering problems like tension spring design, three-bar truss design, and others, DTSA achieved optimal values, as documented in Tables 11–15, indicating its practical utility and robustness in solving constrained optimization tasks.

These outcomes and comparisons solidify DTSA's scientific contribution by offering a quantifiable advancement in optimization efficiency, accuracy, and versatility across a broad spectrum of problem domains.

6. Research Constraints and Considerations

While DTSA has undeniably demonstrated significant enhancements in its optimization capabilities, particularly excelling in addressing high-dimensional and multimodal optimization problems, there remain several aspects warranting further exploration and refinement to augment its versatility and practical utility. These areas include:

Firstly, the algorithm's universality and generalization capacity: Despite impressive performances on numerous IEEE CEC 2014 benchmark functions, DTSA's advancements are less pronounced in scenarios involving simple structures or unimodal functions. This suggests a need for refining the algorithm to better accommodate diverse problem characteristics, encompassing low-dimensional and straightforward optimization scenarios, thus elevating its generalization prowess.

Secondly, parameter tuning and sensitivity: DTSA's performance is significantly influenced by meticulous parameter calibration. Presently, research on parameter selection and tuning strategies might not delve deep enough, necessitating a more systematic analysis of parameter sensitivity. This would facilitate the establishment of default or adaptive parameter settings applicable to a broader spectrum of problems, alleviating the user's burden in manual parameter adjustments.

Lastly, computational efficiency for large-scale problems: As problem dimensions escalate, so do DTSA's computational costs and memory requirements, potentially impeding its application to ultra-large-scale optimization tasks. Advancements in search strategies and data structures, or the incorporation of distributed computing frameworks to parallelize the algorithm, emerge as promising avenues to enhance the efficiency of tackling such massive problems.

To facilitate further research and practical application, we have made the source code publicly accessible at www.jianhuajiang.com (accessed on 1 June 2024), enabling the community to engage with and enhance the algorithm.

Author Contributions: J.J. and J.H. conceived and designed the methodology and experiments; J.H. performed the experiments, analyzed the results and wrote the paper; J.H., J.W., J.L., X.Y. and W.L. revised the paper. All authors have read and agreed to the published version of the manuscript.

Funding: The authors thank the financial support from the Foundation of the Jilin Provincial Department of Science and Technology (No.YDZJ202201ZYTS565), the Foundation of Social Science of Jilin Province, China (No.2022B84) and the Jilin Provincial Department of Education Science and Technology (No.JJKH20240198KJ)

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by Simulated Annealing. *Science* **1983**, *220*, 671–680.
2. Homaifar, A.; Qi, C.X.; Lai, S.H. Constrained optimization via genetic algorithms. *Simulation* **1994**, *62*, 242–253.
3. Deb, K. An efficient constraint handling method for genetic algorithms. *Comput. Methods Appl. Mech. Eng.* **2000**, *186*, 311–338.
4. Liao, T.W. Two hybrid differential evolution algorithms for engineering design optimization. *Appl. Soft Comput.* **2010**, *10*, 1188–1199.
5. Yildiz, K.; Lesieutre, G.A. Sizing and prestress optimization of Class-2 tensegrity structures for space boom applications. *Eng. Comput.* **2022**, *38*, 1451–1464.
6. Osaba, E.; Villar-Rodriguez, E.; Del Ser, J.; Nebro, A.J.; Molina, D.; Latorre, A.; Suganthan, P.N.; Coello, C.A.C.; Herrera, L.E.F. A Tutorial On the design, experimentation and application of metaheuristic algorithms to real-World optimization problems. *Swarm Evol. Comput.* **2021**, *64*, 100888.
7. Tang, J.; Liu, G.; Pan, Q. A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 1627–1643.

8. Tian, Y.; Si, L.; Zhang, X.; Cheng, R.; He, C., Tan, K.C.; Jin, Y. Evolutionary large-scale multi-objective optimization: A survey. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–34.
9. Lodi, A.; Martello, S.; Vigo, D. Heuristic and metaheuristic approaches for a class of two-dimensional bin packing problems. *INFORMS J. Comput.* **1999**, *11*, 345–357.
10. Li, S.; Chen, H.; Wang, M.; Heidari, A.A.; Mirjalili, S. Slime mould algorithm: A new method for stochastic optimization. *Future Gener. Comput. Syst.* **2020**, *111*, 300–323.
11. Hussain, K.; Mohd Salleh, M.N.; Cheng, S.; Shi, Y. Metaheuristic research: A comprehensive survey. *Artif. Intell. Rev.* **2019**, *52*, 2191–2233.
12. Gharehchopogh, F.S. Quantum-inspired metaheuristic algorithms: Comprehensive survey and classification. *Artif. Intell. Rev.* **2023**, *56*, 5479–5543.
13. Braik, M.; Sheta, A.; Al-Hiary, H. A novel meta-heuristic search algorithm for solving optimization problems: Capuchin search algorithm. *Neural Comput. Appl.* **2021**, *33*, 2515–2547.
14. Jiang, J.; Wu, J.; Luo, J.; Yang, X.; Huang, Z. MOBCA: Multi-Objective Besiege and Conquer Algorithm. *Biomimetics* **2024**, *9*, 316.
15. Parejo, J.A.; Ruiz-Cortés, A.; Lozano, S.; Fernandez, P. Metaheuristic optimization frameworks: A survey and benchmarking. *Soft Comput.* **2012**, *16*, 527–561.
16. Abualigah, L. Group search optimizer: A nature-inspired meta-heuristic optimization algorithm with its results, variants, and applications. *Neural Comput. Appl.* **2021**, *33*, 2949–2972.
17. Bai, J.; Jia, L.; Peng, Z. A new insight on augmented Lagrangian method with applications in machine learning. *J. Sci. Comput.* **2024**, *99*, 53.
18. Lin, Y.; Bian, Z.; Liu, X. Developing a dynamic neighborhood structure for an adaptive hybrid simulated annealing–tabu search algorithm to solve the symmetrical traveling salesman problem. *Appl. Soft Comput.* **2016**, *49*, 937–952.
19. Xue, X.; Chen, J. Using compact evolutionary tabu search algorithm for matching sensor ontologies. *Swarm Evol. Comput.* **2019**, *48*, 25–30.
20. Li, J.; Pardalos, P.M.; Sun, H.; Pei, J.; Zhang, Y. Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups. *Expert Syst. Appl.* **2015**, *42*, 3551–3561.
21. Derbel, H.; Jarboui, B.; Hanafi, S.; Chabchoub, H. Genetic algorithm with iterated local search for solving a location-routing problem. *Expert Syst. Appl.* **2012**, *39*, 2865–2871.
22. Vrugt, J.A.; Robinson, B.A.; Hyman, J.M. Self-adaptive multimethod search for global optimization in real-parameter spaces. *IEEE Trans. Evol. Comput.* **2008**, *13*, 243–259.
23. Deng, W.; Xu, J.; Song, Y.; Zhao, H. Differential evolution algorithm with wavelet basis function and optimal mutation strategy for complex optimization problem. *Appl. Soft Comput.* **2021**, *100*, 106724.
24. Das, S.; Abraham, A.; Chakraborty, U.K.; Konar, A. Differential evolution using a neighborhood-based mutation operator. *IEEE Trans. Evol. Comput.* **2009**, *13*, 526–553.
25. Wang, D.; Tan, D.; Liu, L. Particle swarm optimization algorithm: An overview. *Soft Comput.* **2018**, *22*, 387–408.
26. Zhang, Y.; Wang, S.; Ji, G. A comprehensive survey on particle swarm optimization algorithm and its applications. *Math. Probl. Eng.* **2015**, *2015*, 931256.
27. Kıran, M.S.; Fındık, O. A directed artificial bee colony algorithm. *Appl. Soft Comput.* **2015**, *26*, 454–462.
28. Xue, Y.; Jiang, J.; Zhao, B.; Ma, T. A self-adaptive artificial bee colony algorithm based on global best for global optimization. *Soft Comput.* **2018**, *22*, 2935–2952.
29. Gandomi, A.H.; Alavi, A.H. Krill herd: A new bio-inspired optimization algorithm. *Commun. Nonlinear Sci. Numer. Simul.* **2012**, *17*, 4831–4845.
30. Bolaji, A.L.; Al-Betar, M.A.; Awadallah, M.A.; Khader, A.T.; Abualigah, L.M. A comprehensive review: Krill Herd algorithm (KH) and its applications. *Appl. Soft Comput.* **2016**, *49*, 437–446.
31. Rashedi, E.; Rashedi, E.; Nezamabadi-Pour, H. A comprehensive survey on gravitational search algorithm. *Swarm Evol. Comput.* **2018**, *41*, 141–158.
32. Taradeh, M.; Mafarja, M.; Heidari, A.A.; Faris, H.; Aljarah, I.; Mirjalili, S.; Fujita, H. An evolutionary gravitational search-based feature selection. *Inf. Sci.* **2019**, *497*, 219–239.
33. MiarNaeimi, F.; Azizyan, G.; Rashki, M. Horse herd optimization algorithm: A nature-inspired algorithm for high-dimensional optimization problems. *Knowl.-Based Syst.* **2021**, *213*, 106711.
34. Kiran, M.S. TSA: Tree-seed algorithm for continuous optimization. *Expert Syst. Appl.* **2015**, *42*, 6686–6698.
35. El-Fergany, A.A.; Hasanien, H.M. Tree-seed algorithm for solving optimal power flow problem in large-scale power systems incorporating validations and comparisons. *Appl. Soft Comput.* **2018**, *64*, 307–316.
36. Jiang, J.; Yang, X.; Li, M.; Chen, T. ATSA: An Adaptive Tree Seed Algorithm based on double-layer framework with tree migration and seed intelligent generation. *Knowl.-Based Syst.* **2023**, *279*, 110940.
37. Jiang, J.; Wu, J.; Meng, X.; Qian, L.; Luo, J.; Li, K. Katsa: Knn Ameliorated Tree-Seed Algorithm for Complex Optimization Problems. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4636664 (accessed on 1 June 2024).
38. Jiang, J.; Meng, X.; Qian, L.; Wang, H. Enhance tree-seed algorithm using hierarchy mechanism for constrained optimization problems. *Expert Syst. Appl.* **2022**, *209*, 118311.

39. Beşkirli, M. Solving continuous optimization problems using the tree seed algorithm developed with the roulette wheel strategy. *Expert Syst. Appl.* **2021**, *170*, 114579.
40. Beşkirli, A.; Özdemir, D.; Temurtaş, H. A comparison of modified tree–seed algorithm for high-dimensional numerical functions. *Neural Comput. Appl.* **2020**, *32*, 6877–6911.
41. Caponetto, R.; Fortuna, L.; Fazzino, S.; Xibilia, M.G. Chaotic sequences to improve the performance of evolutionary algorithms. *IEEE Trans. Evol. Comput.* **2003**, *7*, 289–304.
42. Jiang, J.; Xu, M.; Meng, X.; Li, K. STSA: A sine Tree-Seed Algorithm for complex continuous optimization problems. *Phys. A Stat. Mech. Appl.* **2020**, *537*, 122802.
43. Bajer, D.; Martinović, G.; Brest, J. A population initialization method for evolutionary algorithms based on clustering and Cauchy deviates. *Expert Syst. Appl.* **2016**, *60*, 294–310.
44. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82.
45. Zheng, Y.; Li, L.; Qian, L.; Cheng, B.; Hou, W.; Zhuang, Y. Sine-SSA-BP ship trajectory prediction based on chaotic mapping improved sparrow search algorithm. *Sensors* **2023**, *23*, 704.
46. Beşkirli, M.; Kiran, M.S. Optimization of Butterworth and Bessel Filter Parameters with Improved Tree-Seed Algorithm. *Biomimetics* **2023**, *8*, 540.
47. Jiang, J.; Liu, Y.; Zhao, Z. TriTSA: Triple Tree-Seed Algorithm for dimensional continuous optimization and constrained engineering problems. *Eng. Appl. Artif. Intell.* **2021**, *104*, 104303.
48. Jiang, J.; Han, R.; Meng, X.; Li, K. TSASC: Tree–seed algorithm with sine–cosine enhancement for continuous optimization problems. *Soft Comput.* **2020**, *24*, 18627–18646.
49. Linden, A. ITSA: Stata Module to Perform Interrupted Time Series Analysis for Single and Multiple Groups. 2021. Available online: https://ideas.repec.org/c/boc/bocode/s457793.html (accessed on 1 June 2024)
50. Jiang, J.; Jiang, S.; Meng, X.; Qiu, C. EST-TSA: An effective search tendency based to tree seed algorithm. *Phys. A Stat. Mech. Appl.* **2019**, *534*, 122323.
51. Jiang, J.; Meng, X.; Chen, Y.; Qiu, C.; Liu, Y.; Li, K. Enhancing tree-seed algorithm via feed-back mechanism for optimizing continuous problems. *Appl. Soft Comput.* **2020**, *92*, 106314.
52. Chen, X.; Przystupa, K.; Ye, Z.; Chen, F.; Wang, C.; Liu, J.; Gao, R.; Wei, M.; Kochan, O. Forecasting short-term electric load using extreme learning machine with improved tree seed algorithm based on Levy flight. *Eksploat. I Niezawodn.* **2022**, *24*, 153–162.
53. Babalik, A.; Cinar, A.C.; Kiran, M.S. A modification of tree-seed algorithm using Deb's rules for constrained optimization. *Appl. Soft Comput.* **2018**, *63*, 289–305.
54. Kanna, S.R.; Sivakumar, K.; Lingaraj, N. Development of deer hunting linked earthworm optimization algorithm for solving large scale traveling salesman problem. *Knowl.-Based Syst.* **2021**, *227*, 107199.
55. Jiang, J.; Meng, X.; Liu, Y.; Wang, H. An enhanced TSA-MLP model for identifying credit default problems. *SAGE Open* **2022**, *12*, 21582440221094586.
56. Aslan, M.F.; Sabanci, K.; Ropelewska, E. A new approach to COVID-19 detection: An ANN proposal optimized through tree-seed algorithm. *Symmetry* **2022**, *14*, 1310.
57. Luo, X.; Chen, J.; Yuan, Y.; Wang, Z. Pseudo Gradient-Adjusted Particle Swarm Optimization for Accurate Adaptive Latent Factor Analysis. *IEEE Trans. Syst. Man Cybern. Syst.* **2024**, *54*, 2213–2226.
58. Ahmad, A.; Yadav, A.K.; Singh, A.; Singh, D.K.; Ağbulut, Ü. A hybrid RSM-GA-PSO approach on optimization of process intensification of linseed biodiesel synthesis using an ultrasonic reactor: Enhancing biodiesel properties and engine characteristics with ternary fuel blends. *Energy* **2024**, *288*, 129077.
59. Nadimi-Shahraki, M.H.; Zamani, H.; Varzaneh, Z.A.; Sadiq, A.S.; Mirjalili, S. A Systematic Review of Applying Grey Wolf Optimizer, its Variants, and its Developments in Different Internet of Things Applications. *Internet Things* **2024**, *26*, 101135.
60. Yılmaz, S.; Küçüksille, E.U. A new modification approach on bat algorithm for solving optimization problems. *Appl. Soft Comput.* **2015**, *28*, 259–275.
61. Ekinci, S.; Izci, D.; Abu Zitar, R.; Alsoud, A.R.; Abualigah, L. Development of Lévy flight-based reptile search algorithm with local search ability for power systems engineering design problems. *Neural Comput. Appl.* **2022**, *34*, 20263–20283.
62. Ajani, O.S.; Kumar, A.; Mallipeddi, R. Covariance matrix adaptation evolution strategy based on correlated evolution paths with application to reinforcement learning. *Expert Syst. Appl.* **2024**, *246*, 123289.
63. Kocak, O.; Erkan, U.; Toktas, A.; Gao, S. PSO-based image encryption scheme using modular integrated logistic exponential map. *Expert Syst. Appl.* **2024**, *237*, 121452.
64. Zheng, Y.; Wang, J.S.; Zhu, J.H.; Zhang, X.Y.; Xing, Y.X.; Zhang, Y.H. MORSA: Multi-objective reptile search algorithm based on elite non-dominated sorting and grid indexing mechanism for wind farm layout optimization problem. *Energy* **2024**, *293*, 130771.
65. He, K.; Zhang, Y.; Wang, Y.K.; Zhou, R.H.; Zhang, H.Z. EABOA: Enhanced adaptive butterfly optimization algorithm for numerical optimization and engineering design problems. *Alex. Eng. J.* **2024**, *87*, 543–573.
66. Doğan, B.; Ölmez, T. A new metaheuristic for numerical function optimization: Vortex Search algorithm. *Inf. Sci.* **2015**, *293*, 125–145.
67. Askarzadeh, A. A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm. *Comput. Struct.* **2016**, *169*, 1–12.

68. Li, Y.; Zhao, Y.; Liu, J. Dimension by dimension dynamic sine cosine algorithm for global optimization problems. *Appl. Soft Comput.* **2021**, *98*, 106933.
69. Russo, I.L.; Bernardino, H.S.; Barbosa, H.J. Knowledge discovery in multiobjective optimization problems in engineering via Genetic Programming. *Expert Syst. Appl.* **2018**, *99*, 93–102.
70. Baghmisheh, M.V.; Peimani, M.; Sadeghi, M.H.; Ettefagh, M.M.; Tabrizi, A.F. A hybrid particle swarm–Nelder–Mead optimization method for crack detection in cantilever beams. *Appl. Soft Comput.* **2012**, *12*, 2217–2226.
71. Gupta, S.; Abderazek, H.; Yıldız, B.S.; Yildiz, A.R.; Mirjalili, S.; Sait, S.M. Comparison of metaheuristic optimization algorithms for solving constrained mechanical design optimization problems. *Expert Syst. Appl.* **2021**, *183*, 115351.